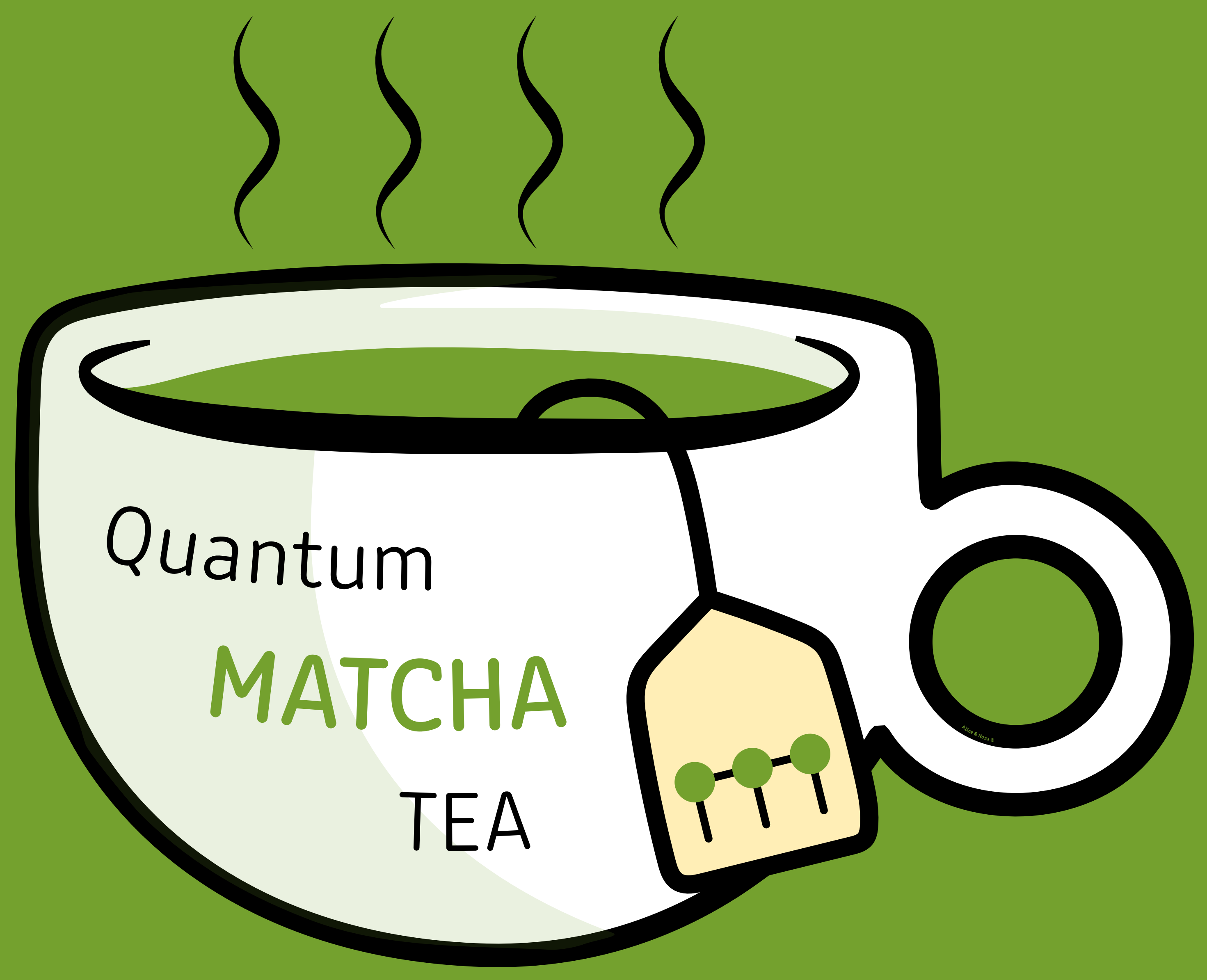
HPCQC - Fifth edition
December 15 2022

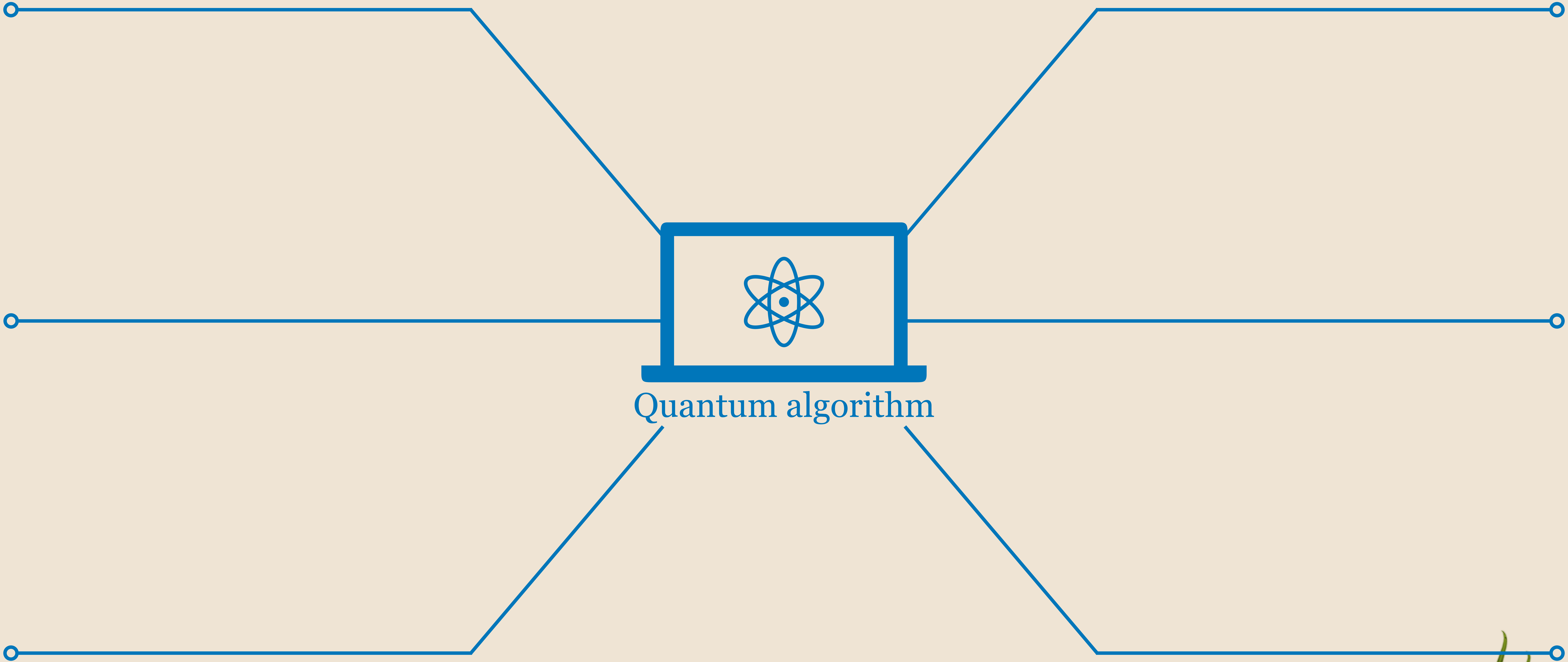
Quantum Matcha Tea

An efficient matrix product state simulator for quantum circuits

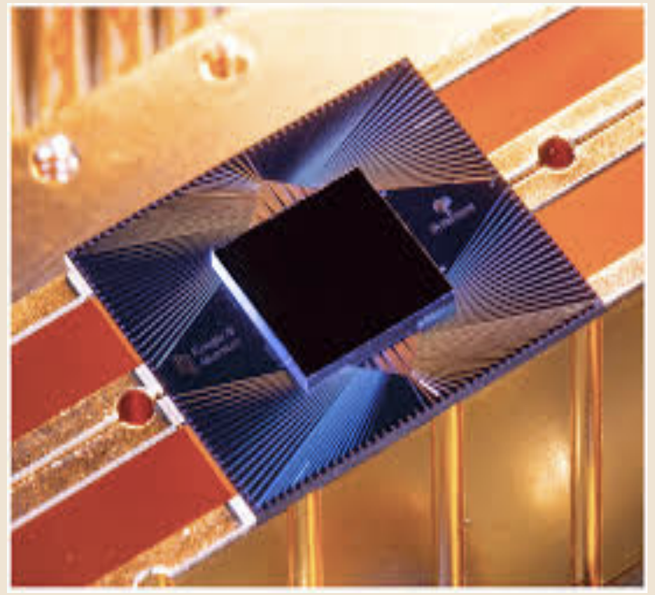


Marco Ballarin
Università degli studi di Padova

Running quantum algorithms



Running quantum algorithms

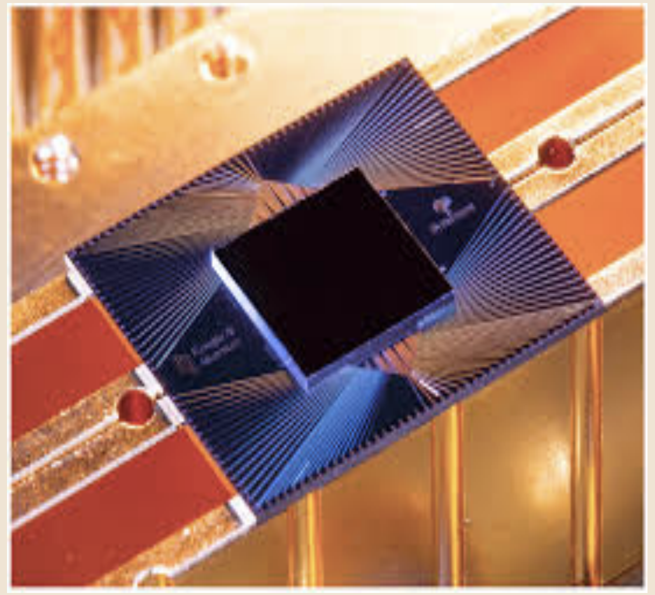


+ Real hardware
- Noisy
- Limited number of qubits

Quantum hardware

Quantum algorithm

Running quantum algorithms



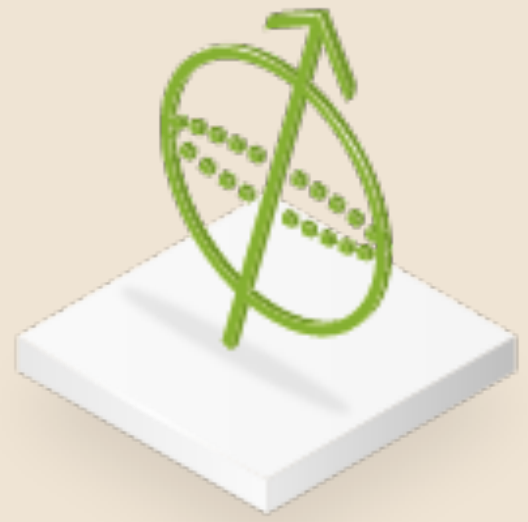
+ Real hardware
- Noisy
- Limited number of qubits

Quantum hardware

High # of qubits +

Flexibility (observables) -

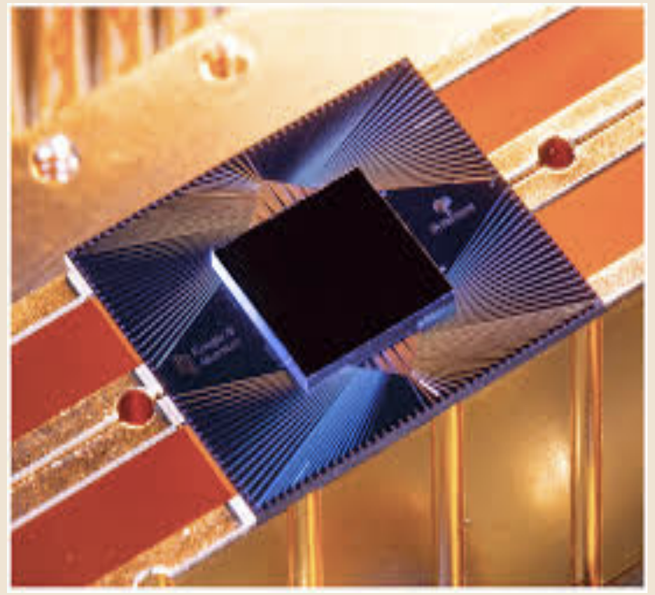
Depth of the circuit -



cuQuantum

Quantum algorithm

Running quantum algorithms



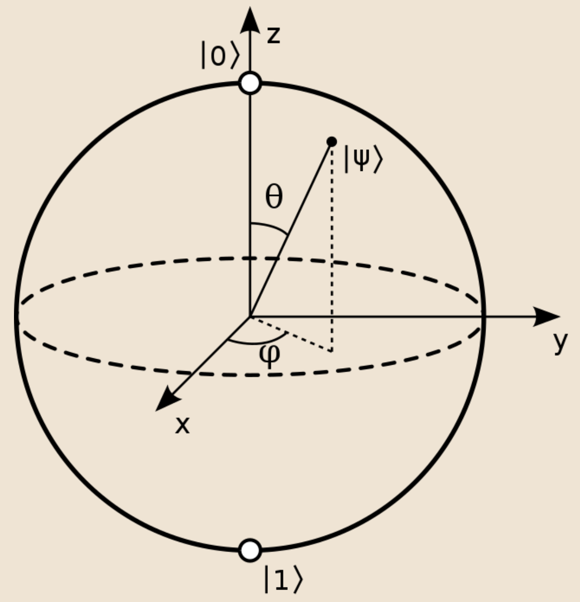
+ Real hardware
- Noisy
- Limited number of qubits

Quantum hardware

High # of qubits +
Flexibility (observables) -
Depth of the circuit -

cuQuantum

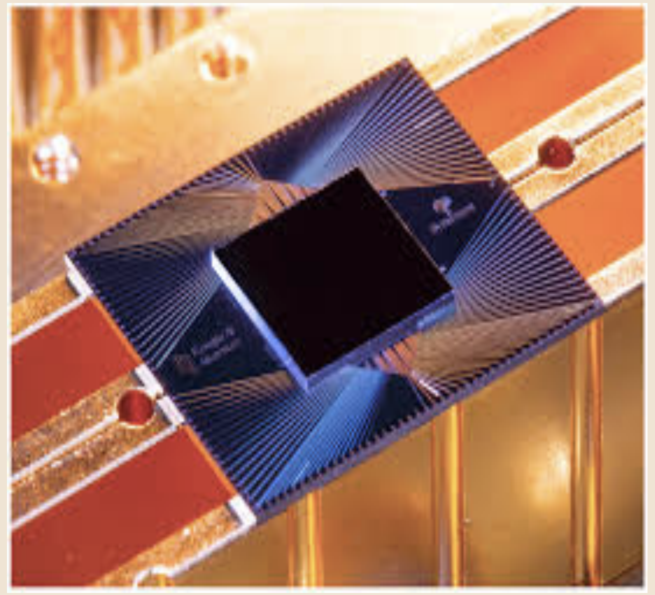
Quantum algorithm



+ Access to exact state
- Limited number of qubits

Exact simulator

Running quantum algorithms



+ Real hardware
- Noisy
- Limited number of qubits

Quantum hardware

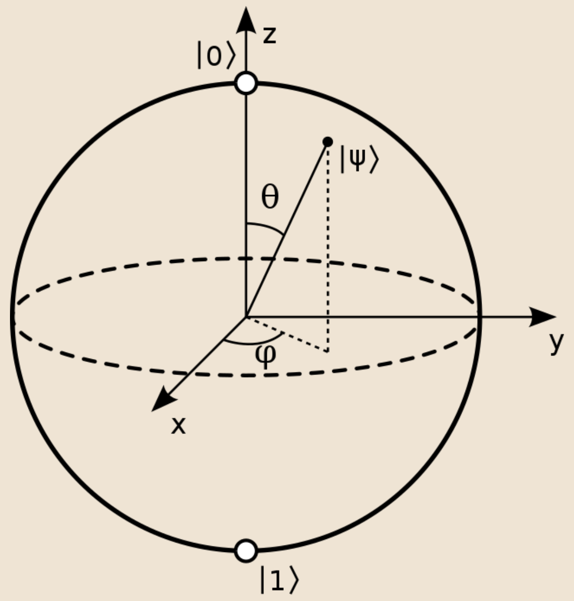
High # of qubits +

Flexibility (observables) -

Depth of the circuit -

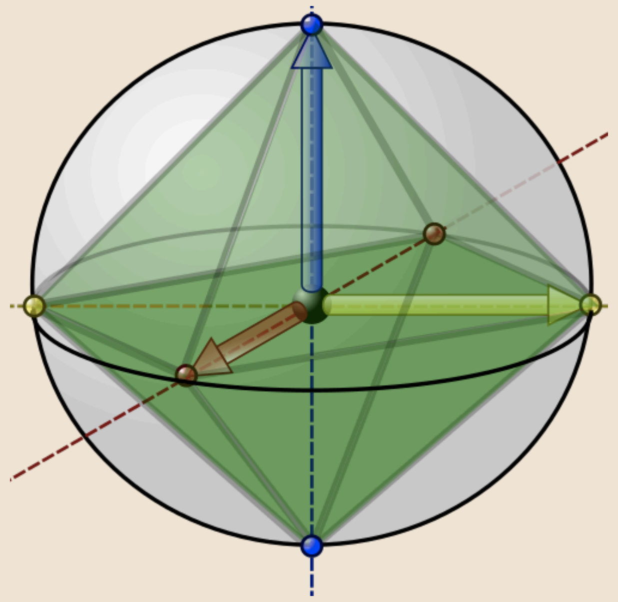
cuQuantum

Quantum algorithm



+ Access to exact state
- Limited number of qubits

Exact simulator

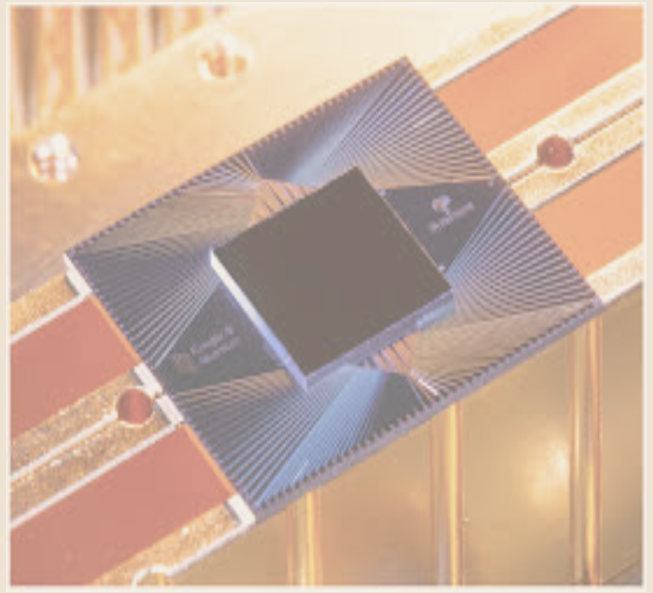


High # of qubits +

Flexibility (# of T gates) -

Clifford simulator

Running quantum algorithms



+ Real hardware
- Noisy
- Limited number of qubits

Quantum hardware

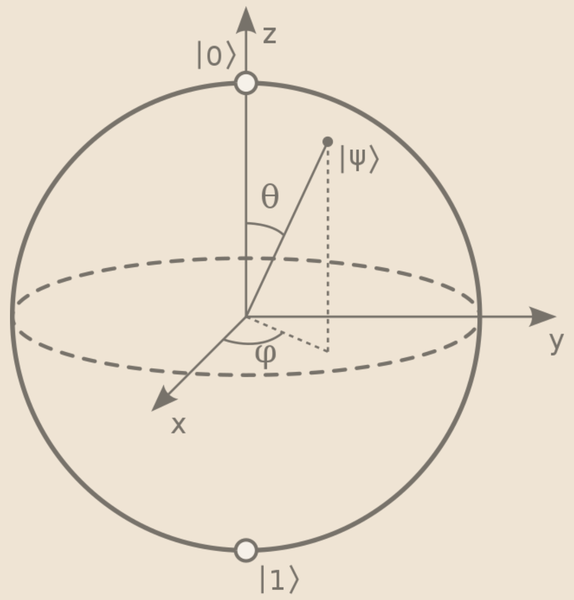
High # of qubits +

Flexibility (observables) -

Depth of the circuit -

cuQuantum

Quantum algorithm

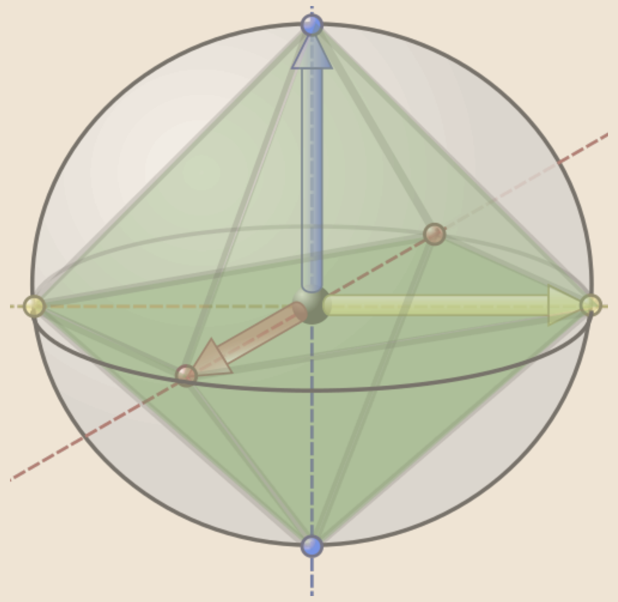


+ Access to exact state
- Limited number of qubits

Exact simulator

High # of qubits +

Flexibility (# of T gates) -



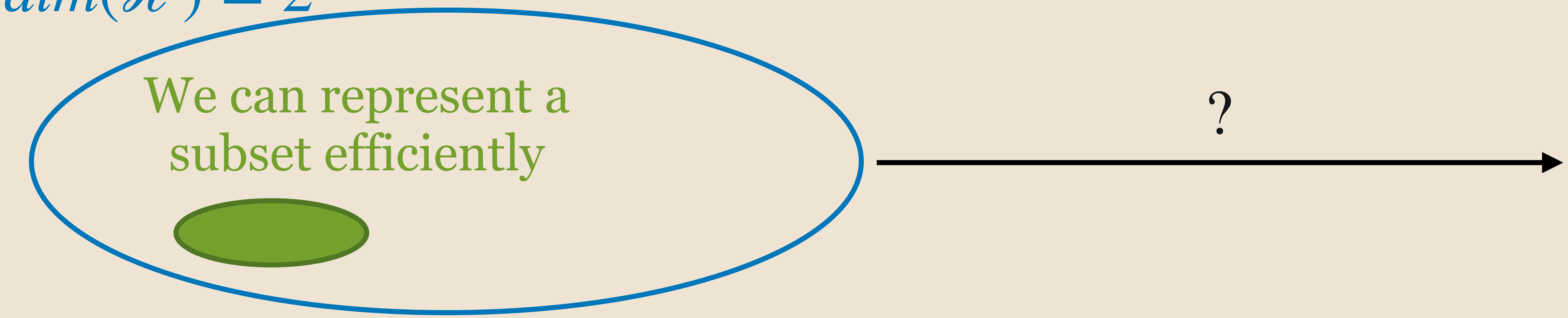
Clifford simulator

Tensor Network simulator

+ High # of qubits
- Flexibility (entanglement)

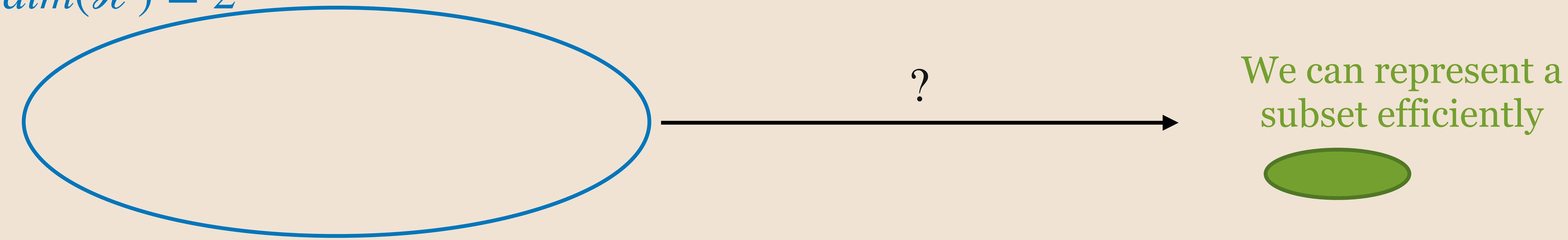
Why tensor networks

$dim(\mathscr{H}) = 2^n$



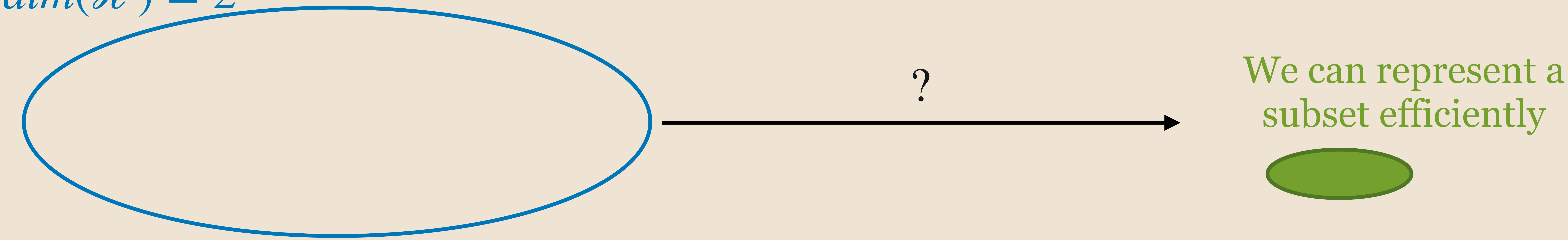
Why tensor networks

$dim(\mathscr{H}) = 2^n$



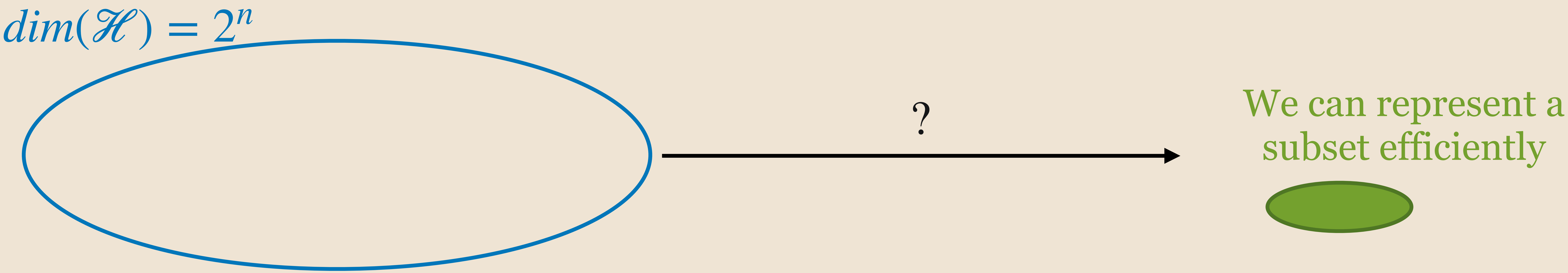
Why tensor networks

$dim(\mathscr{H}) = 2^n$



$$|\psi\rangle = \sum_{\alpha=1}^{\chi} |A_\alpha\rangle \overset{\lambda_\alpha}{—} |B_\alpha\rangle$$

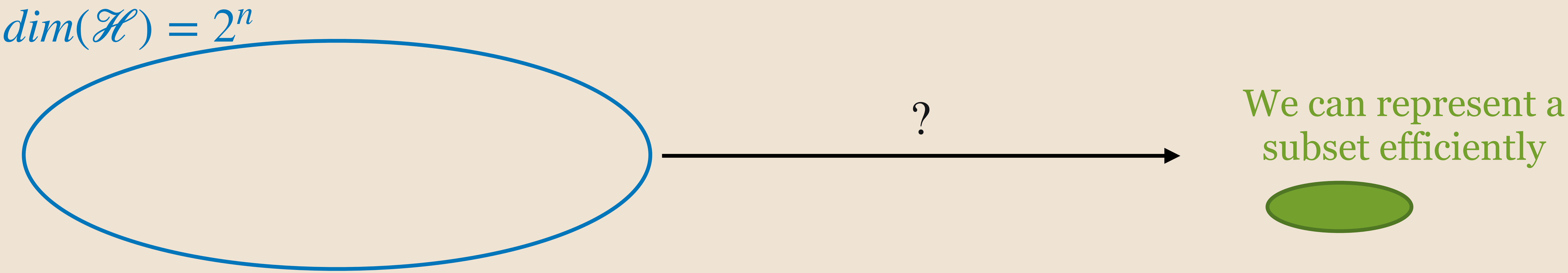
Why tensor networks



Tensor networks compress the quantum correlations between subsystems ⇒ **compress entanglement**

$$|\psi\rangle = \sum_{\alpha=1}^{\chi} |A_\alpha\rangle \, \lambda_\alpha \, |B_\alpha\rangle$$

Why tensor networks

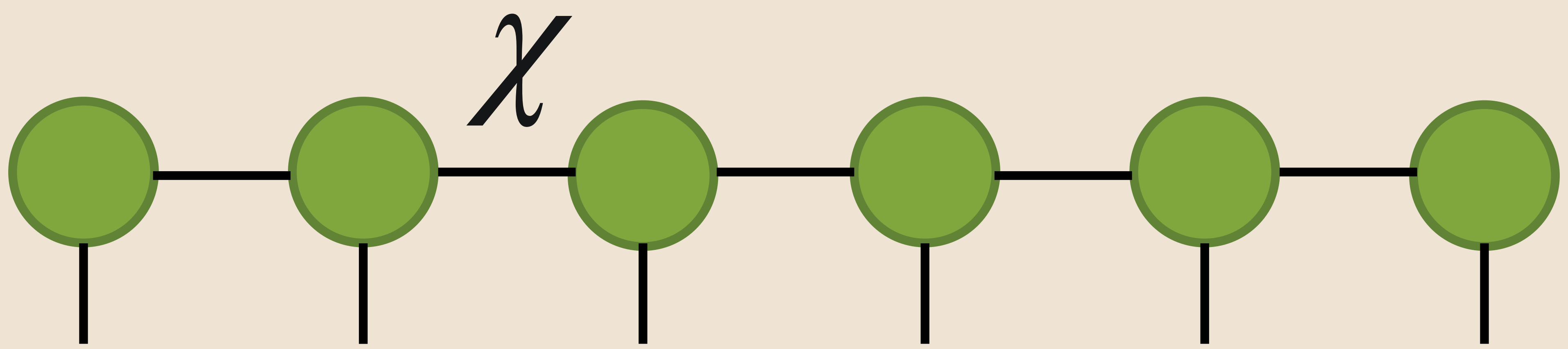


Tensor networks compress the quantum correlations between subsystems ⇒ **compress entanglement**

$$|\psi\rangle = \sum_{\alpha=1}^{\chi} |A_\alpha\rangle \lambda_\alpha |B_\alpha\rangle$$

Only keep highest χ Schmidt values

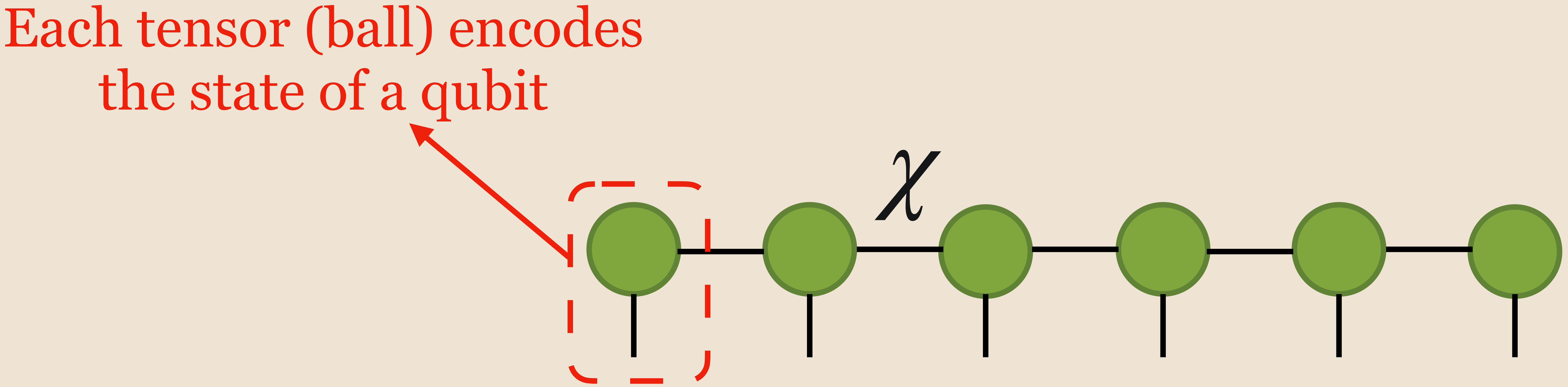
Matrix product states



Memory requirements

$$O(2^n) \rightarrow O(2n\chi^2)$$

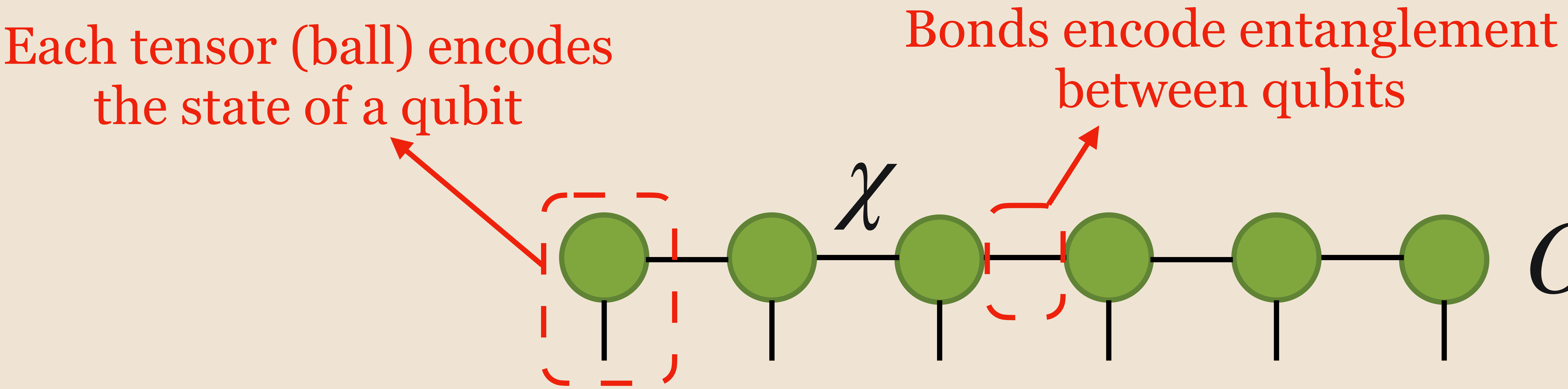
Matrix product states



Memory requirements

$$O(2^n) \rightarrow O(2n\chi^2)$$

Matrix product states



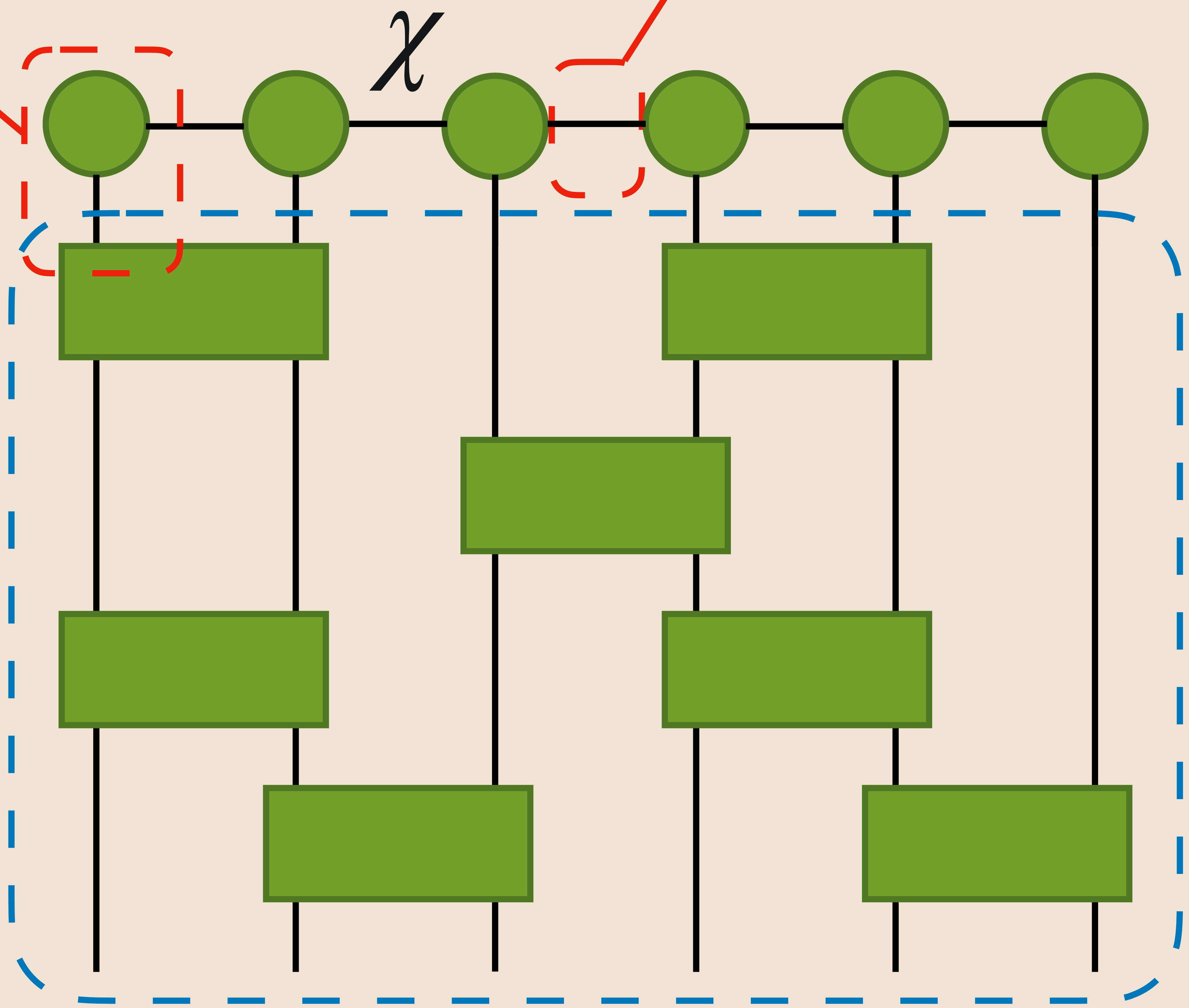
Memory requirements

$$O(2^n) \to O(2n\chi^2)$$

Matrix product states

Each tensor (ball) encodes the state of a qubit

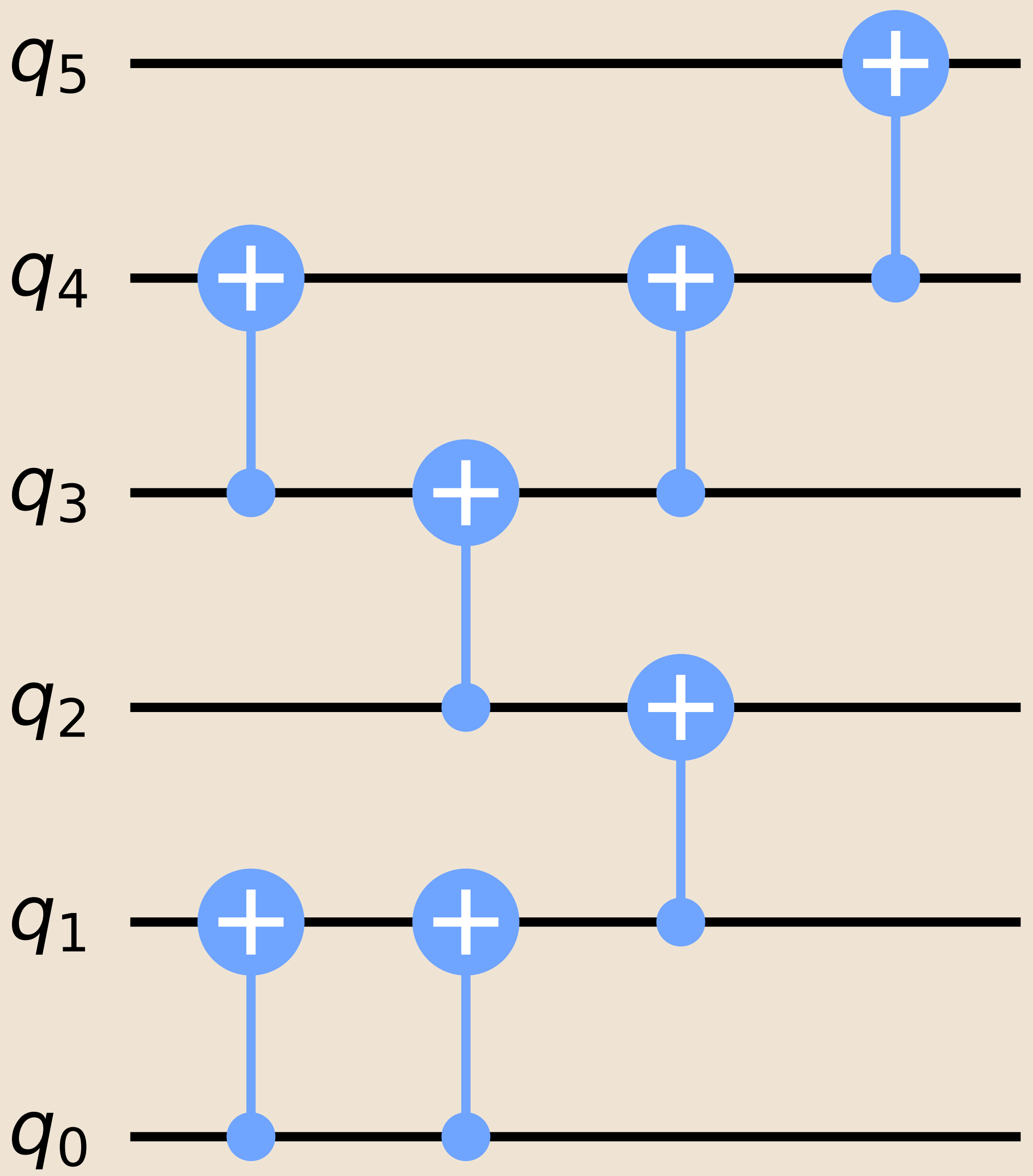
Bonds encode entanglement between qubits



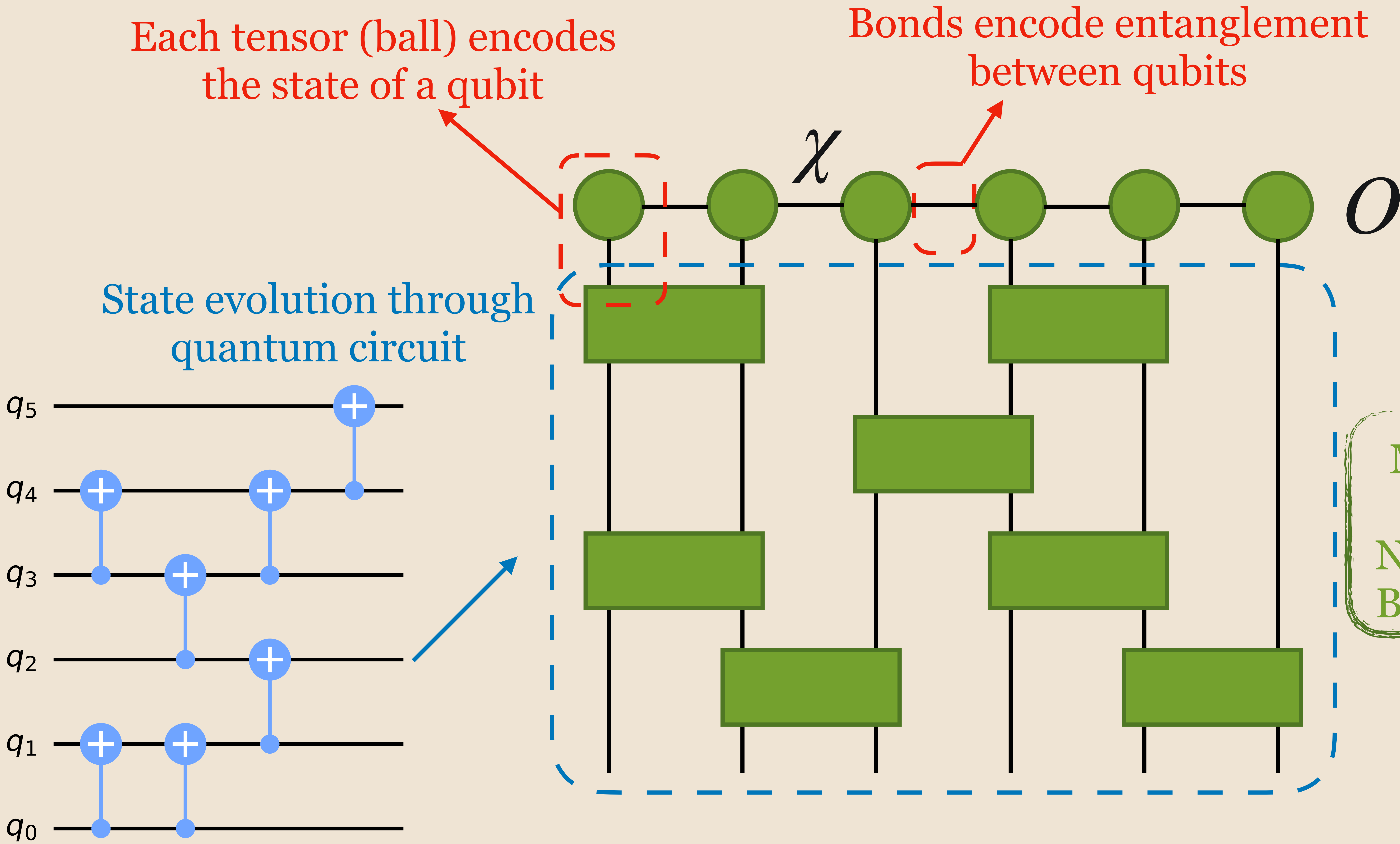
Memory requirements

$$O(2^n) \rightarrow O(2n\chi^2)$$

State evolution through quantum circuit



Matrix product states

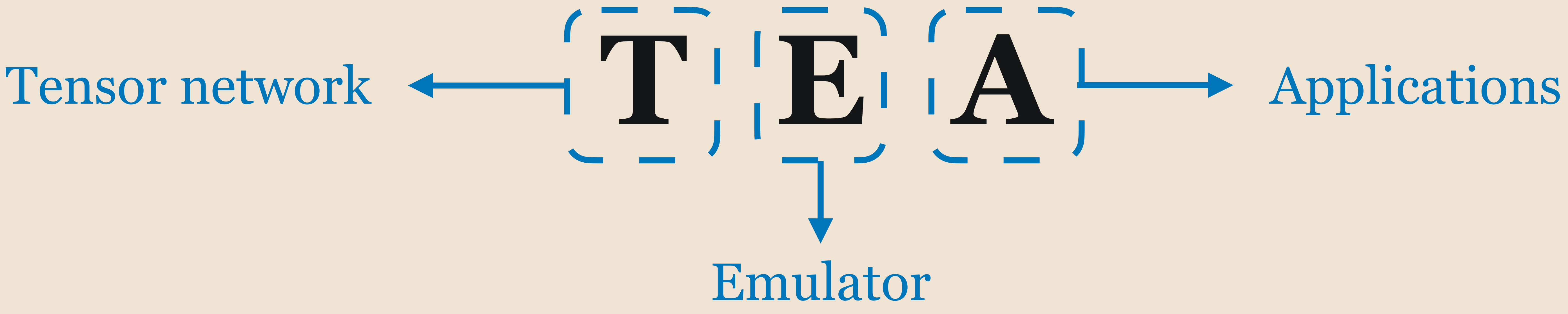


Memory requirements

$$O(2^n) \rightarrow O(2n\chi^2)$$

MPS SIMULATIONS ARE NOT LIMITED BY THE NUMBER OF QUBITS BUT BY THE ENTANGLEMENT

Quantum TEA distribution



Quantum TEA distribution

T E A

- T → Tensor network
- E → Emulator
- A → Applications

Quantum tea leaves: **Utility**

Quantum matcha tea: **quantum circuit HPC simulations**

Quantum red tea: **tensor handling**

Quantum chai tea: **AI and ML with tensor networks**

Quantum green tea: **Schrödinger equation solution for many-body states**

Quantum TEA distribution

Tensor network ← **T** | **E** | **A** → Applications

E ↓ Emulator

Public!

- Quantum tea leaves: **Utility**
- Quantum matcha tea: **quantum circuit HPC simulations**
- Quantum red tea: **tensor handling**
- Quantum chai tea: **AI and ML with tensor networks**
- Quantum green tea: **Schrödinger equation solution for many-body states**

Quantum Matcha Tea workflow

Quantum circuit

Observables

Python interface, definition of the problem

Quantum Matcha Tea workflow

Quantum circuit

Observables

Python interface, definition of the problem

Matrix product state simulator

Quantum Matcha Tea workflow

Quantum circuit

Observables

Python interface, definition of the problem

Matrix product state simulator

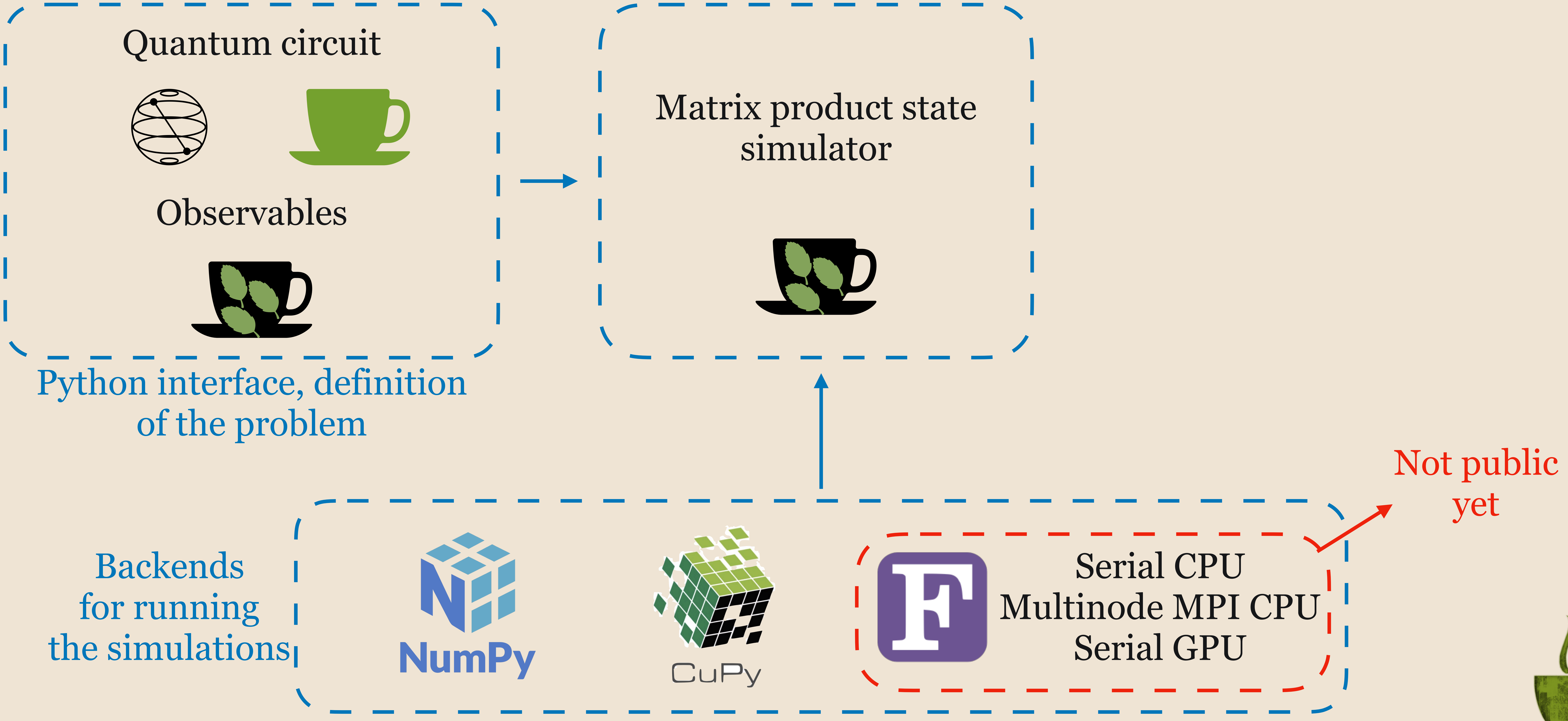
Backends for running the simulations

NumPy

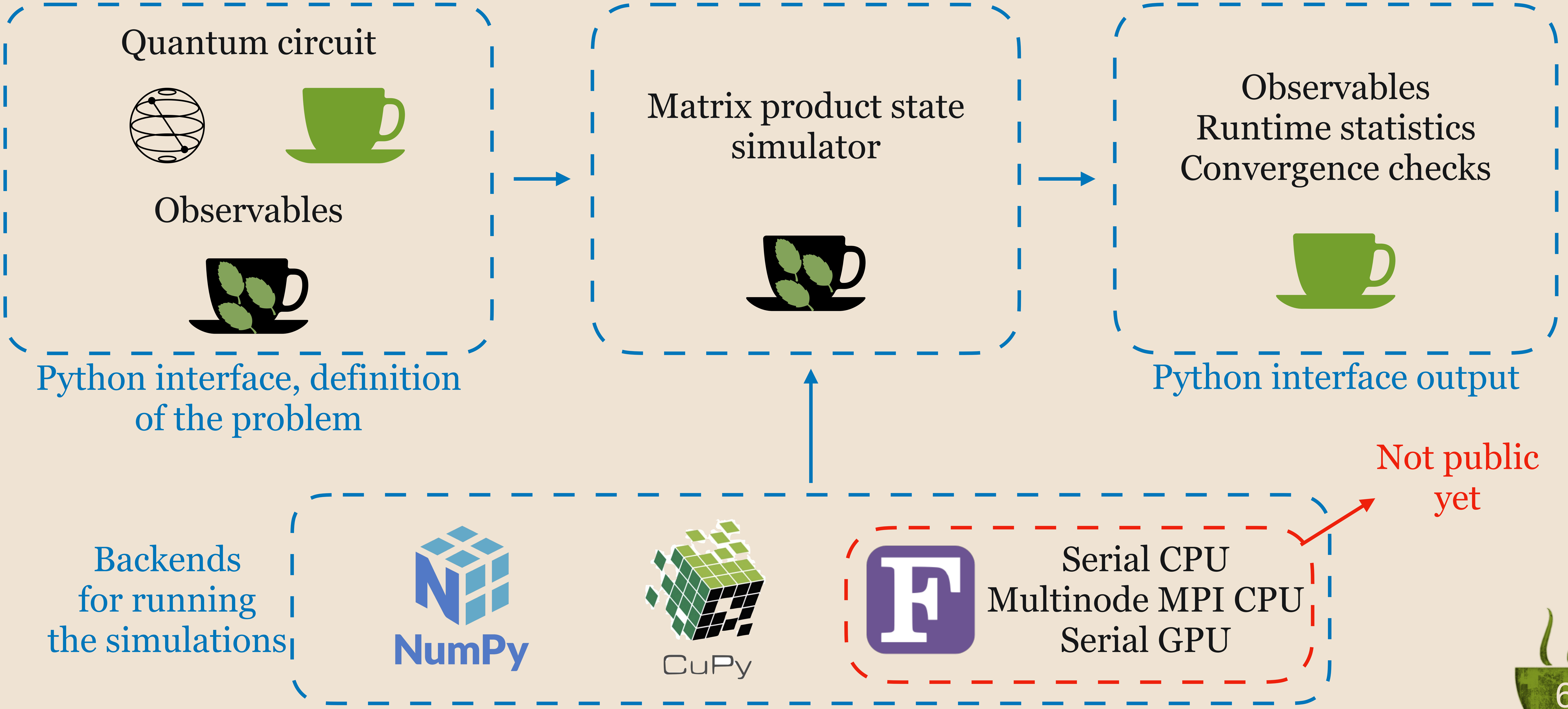
CuPy

Serial CPU
Multinode MPI CPU
Serial GPU

Quantum Matcha Tea workflow



Quantum Matcha Tea workflow

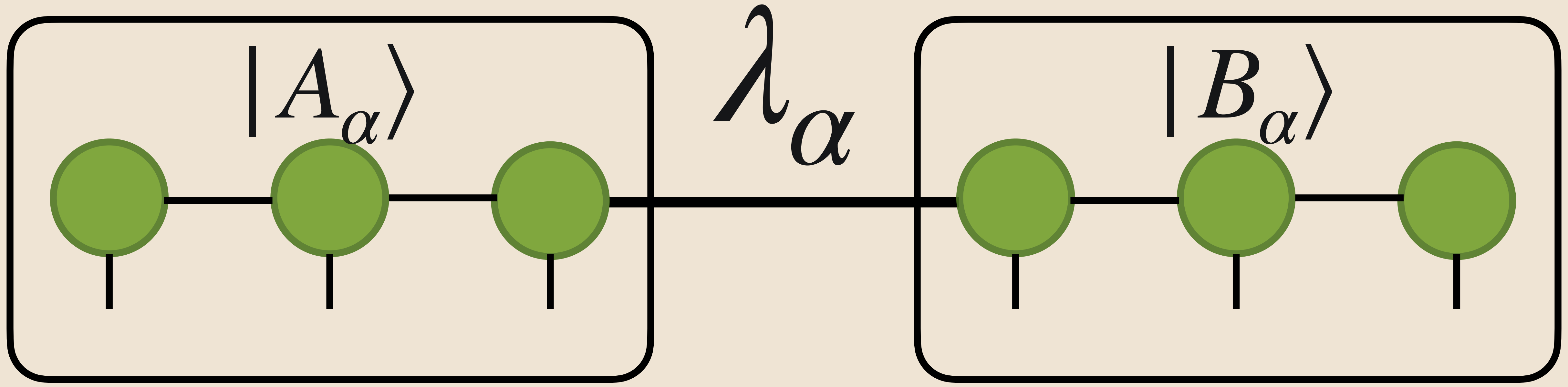


Convergence checks & error bound

$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^{i-1}} |A_\alpha\rangle \overset{\lambda_\alpha}{—} |B_\alpha\rangle$$

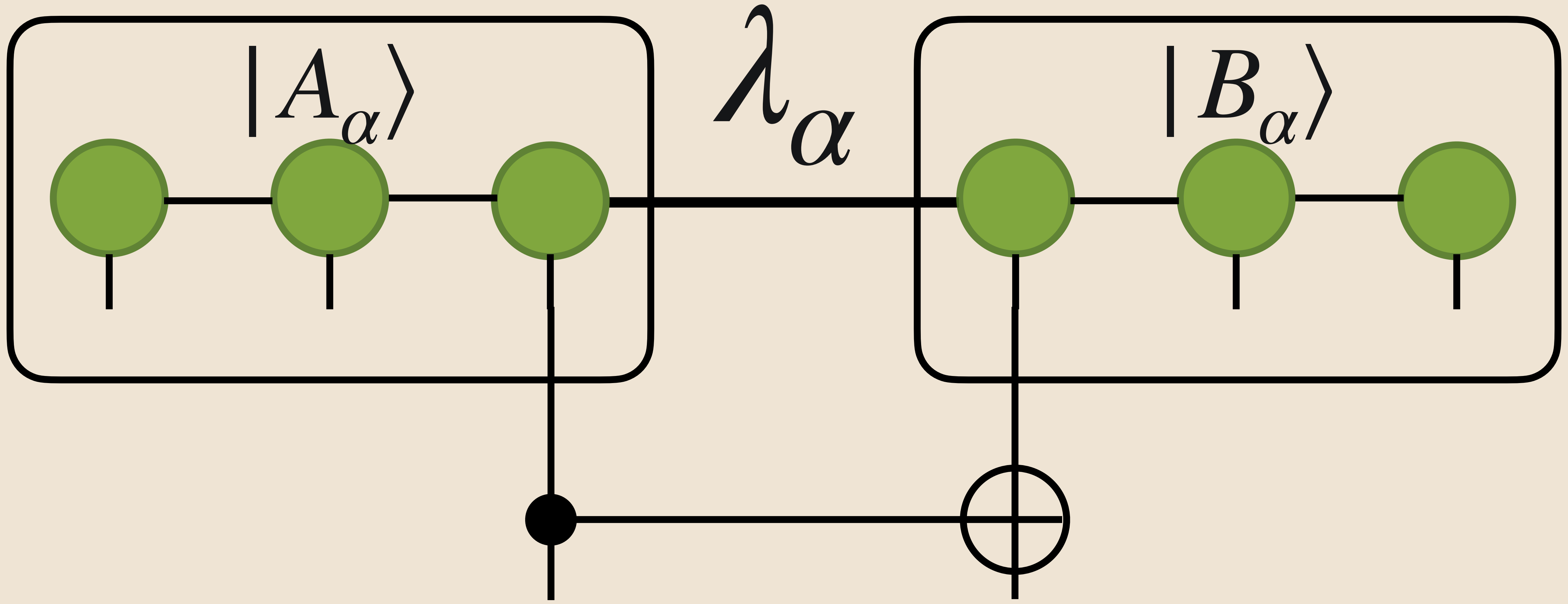
Convergence checks & error bound

$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^{i-1}}$$



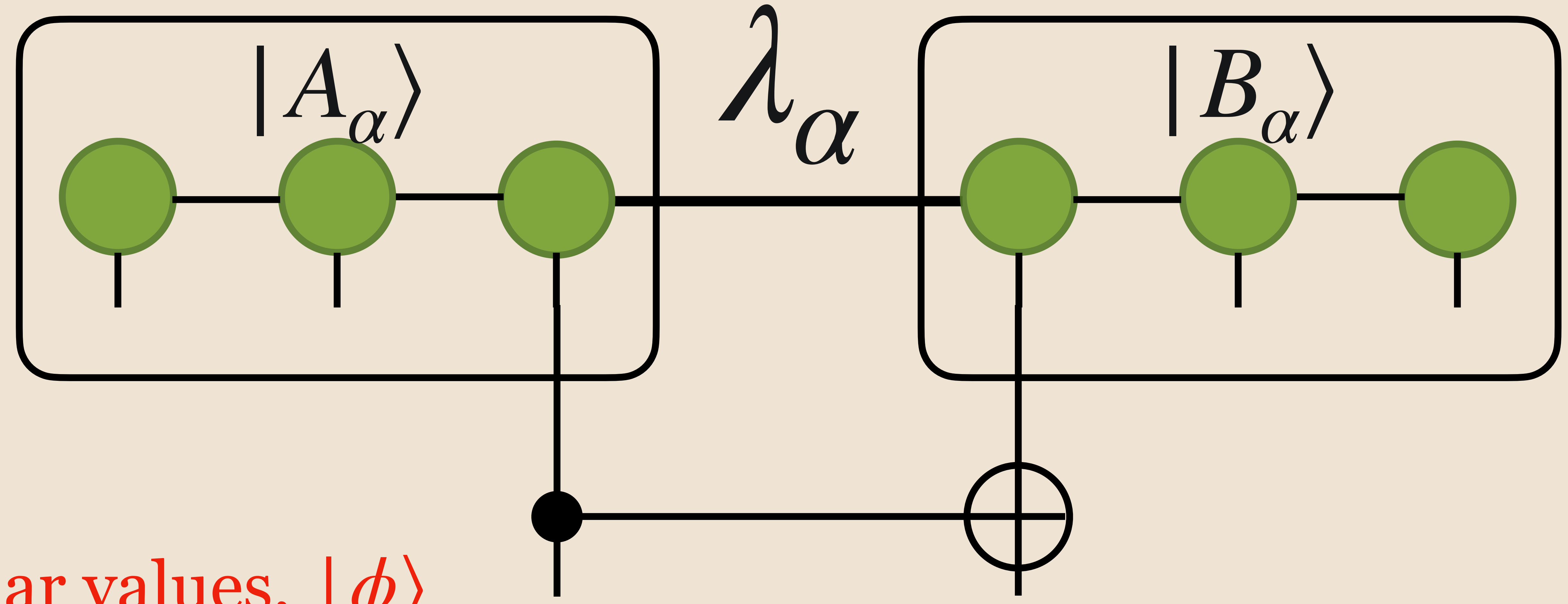
Convergence checks & error bound

$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^{i-1}}$$



Convergence checks & error bound

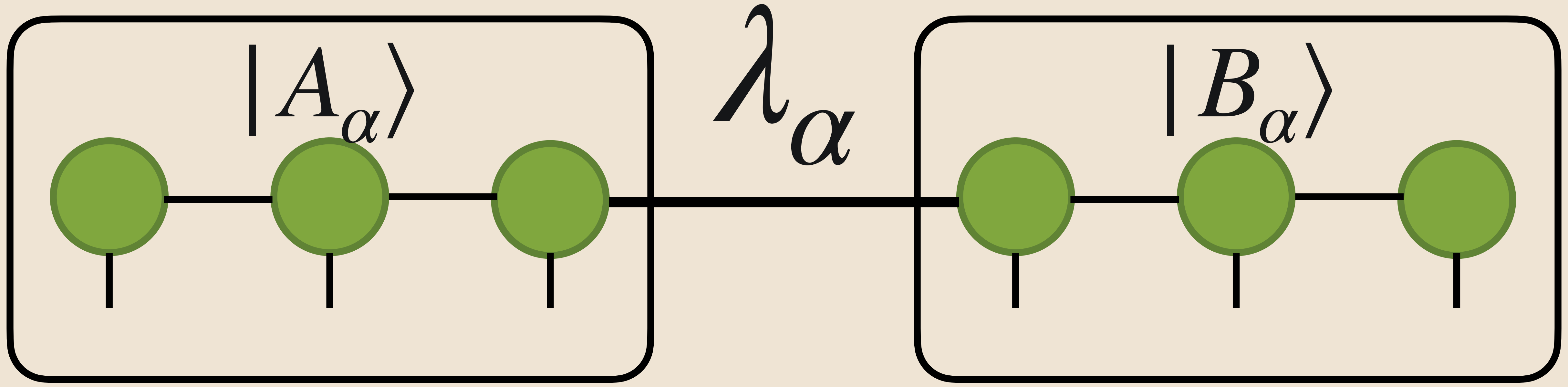
$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^{i-1}}$$



Only keep highest χ singular values, $|\phi\rangle$

Convergence checks & error bound

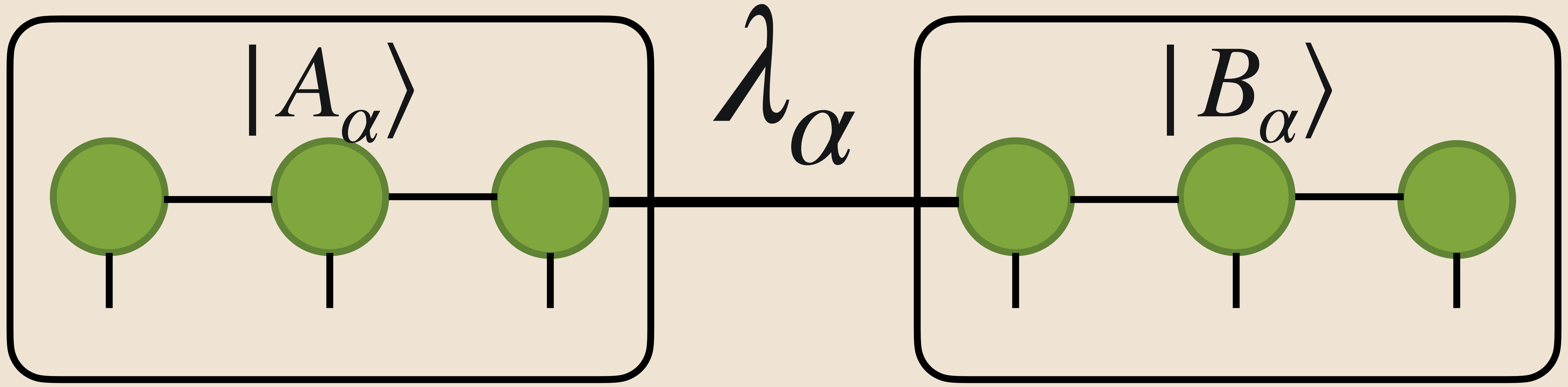
$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^i}$$



Only keep highest χ singular values, $|\phi\rangle$

Convergence checks & error bound

$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^i}$$



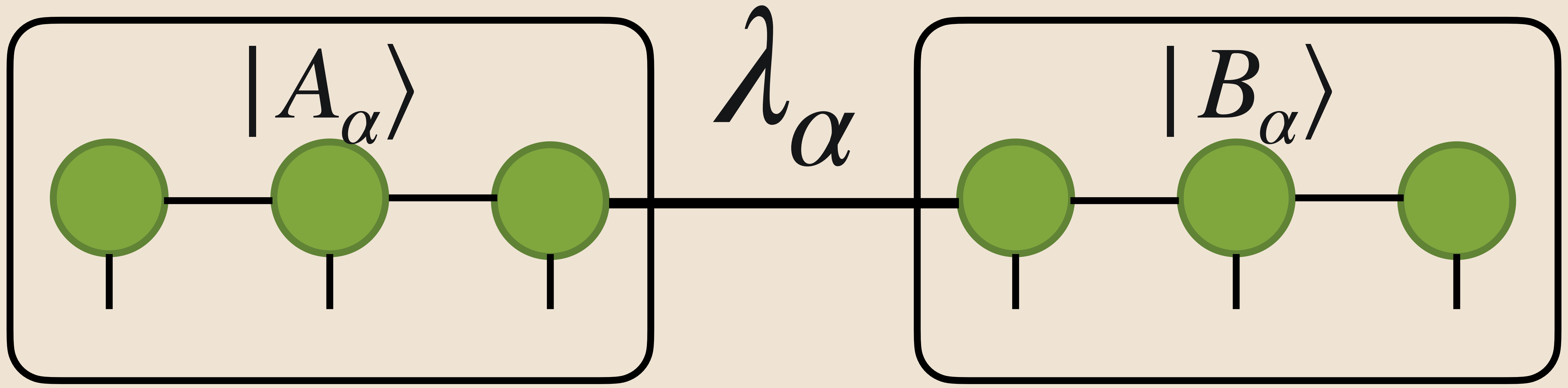
Only keep highest χ singular values, $|\phi\rangle$

Fidelity of the state

$$\mathcal{F}_i(\chi) = \left|\langle\psi|\phi\rangle\right|^2 = \left|1 - \sum_{\alpha=\chi+1}^{\chi_T^i} \lambda_\alpha^2\right|^2$$

Convergence checks & error bound

$$|\psi\rangle = \sum_{\alpha=1}^{\chi_T^i}$$



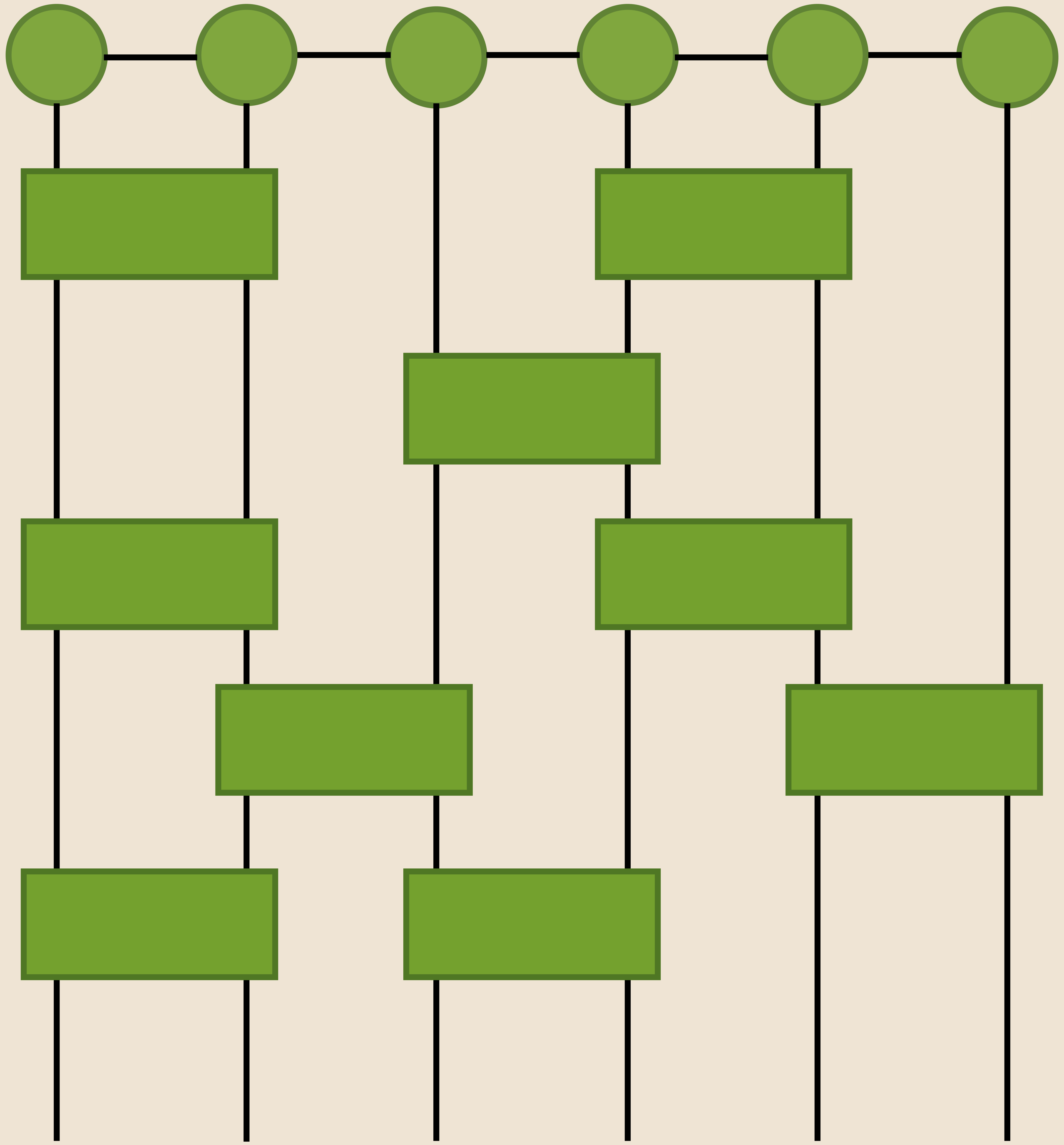
Only keep highest χ singular values, $|\phi\rangle$

Fidelity of the state

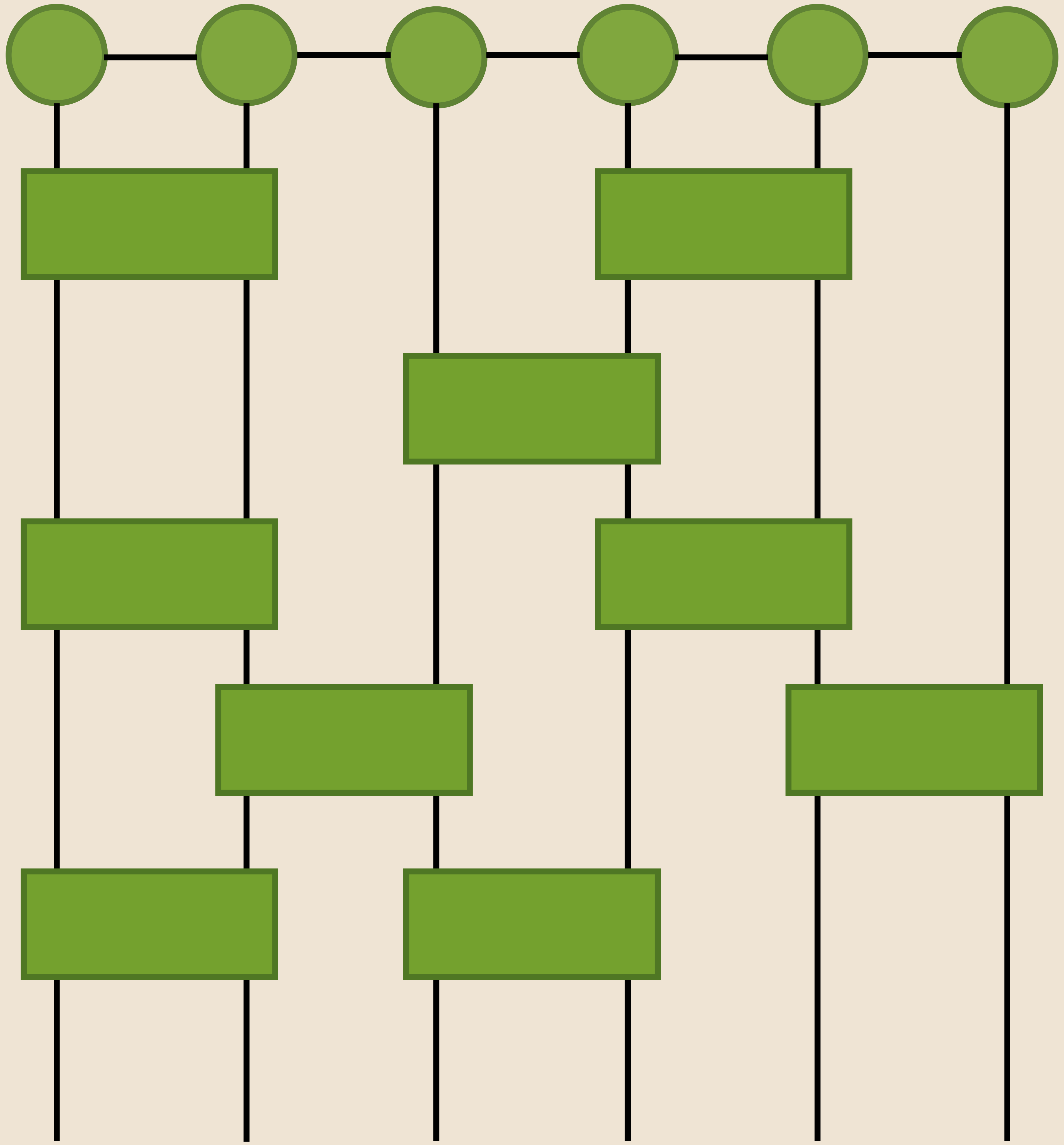
$$\mathcal{F}_i(\chi) = \left|\langle\psi|\phi\rangle\right|^2 = \left|1 - \sum_{\alpha=\chi+1}^{\chi_T^i} \lambda_\alpha^2\right|^2$$

Computed during the simulation

Convergence and error checks



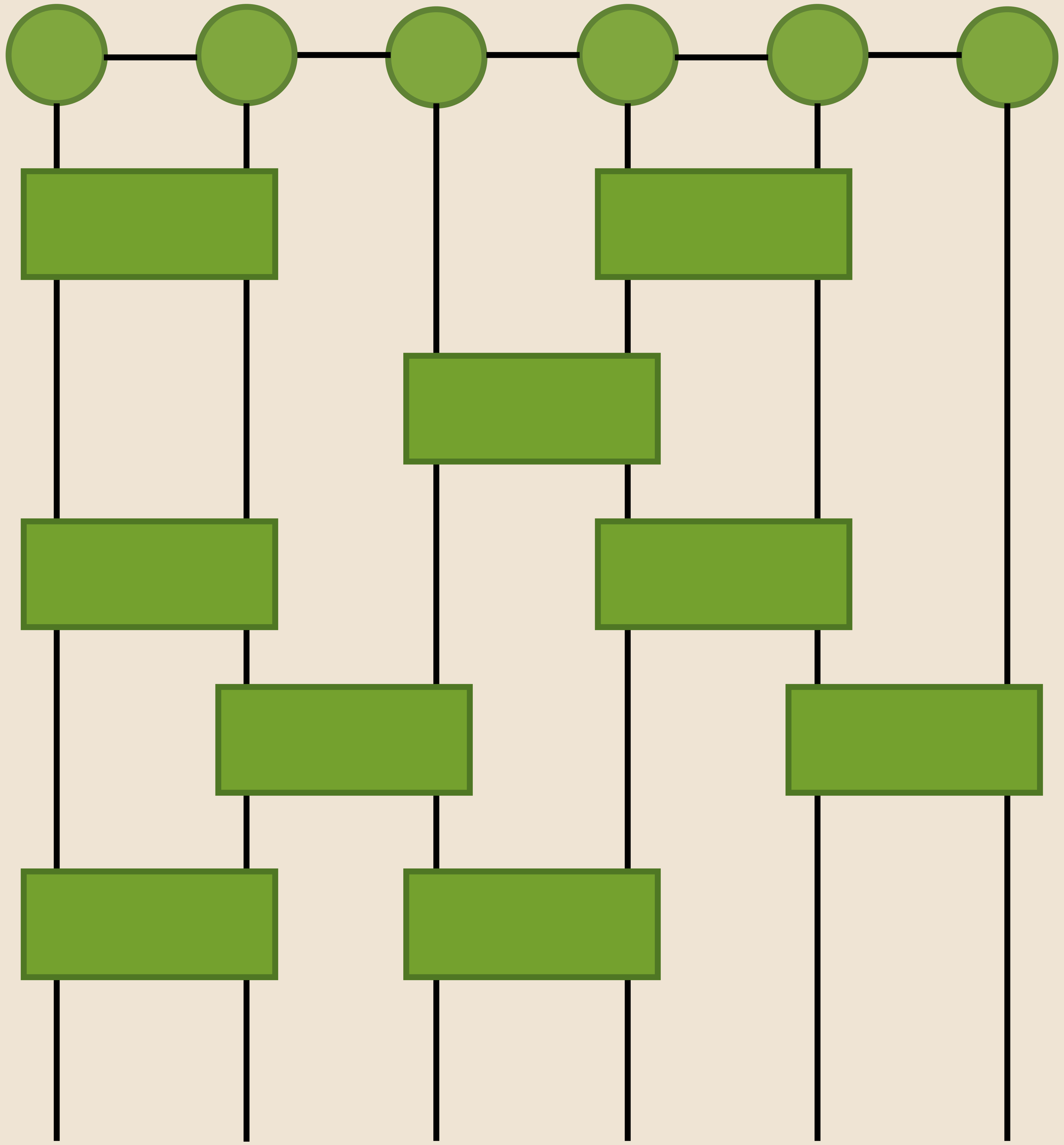
Convergence and error checks



Fidelity of the state after a **single** gate

$$\mathcal{F}_i(\chi) = \left| 1 - \sum_{\alpha=\chi+1}^{\chi_T^i} \lambda_\alpha^2 \right|^2$$

Convergence and error checks



Fidelity of the state after a **single** gate

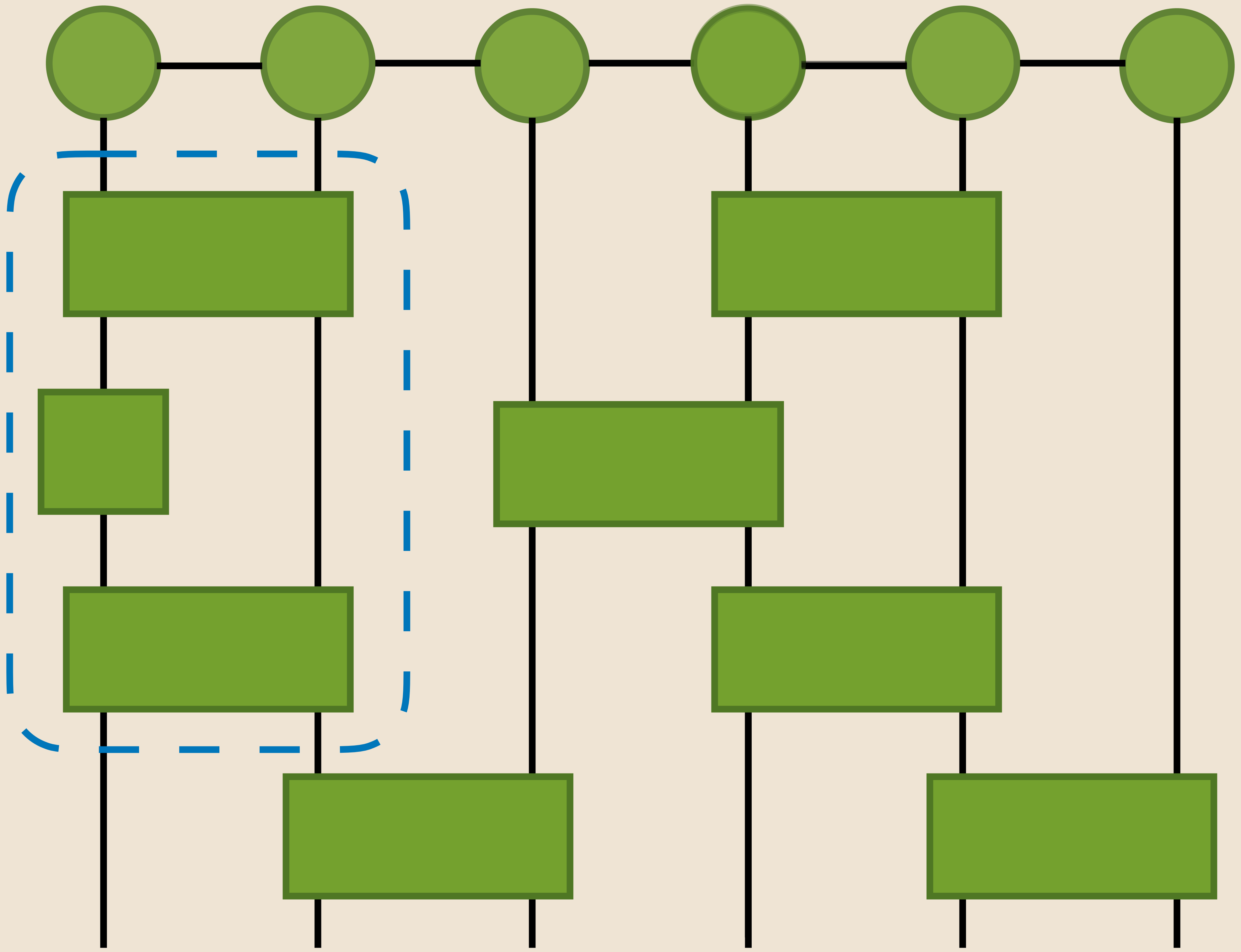
$$\mathcal{F}_i(\chi) = \left| 1 - \sum_{\alpha=\chi+1}^{\chi_T^i} \lambda_\alpha^2 \right|^2$$

Fidelity at the end of the simulation

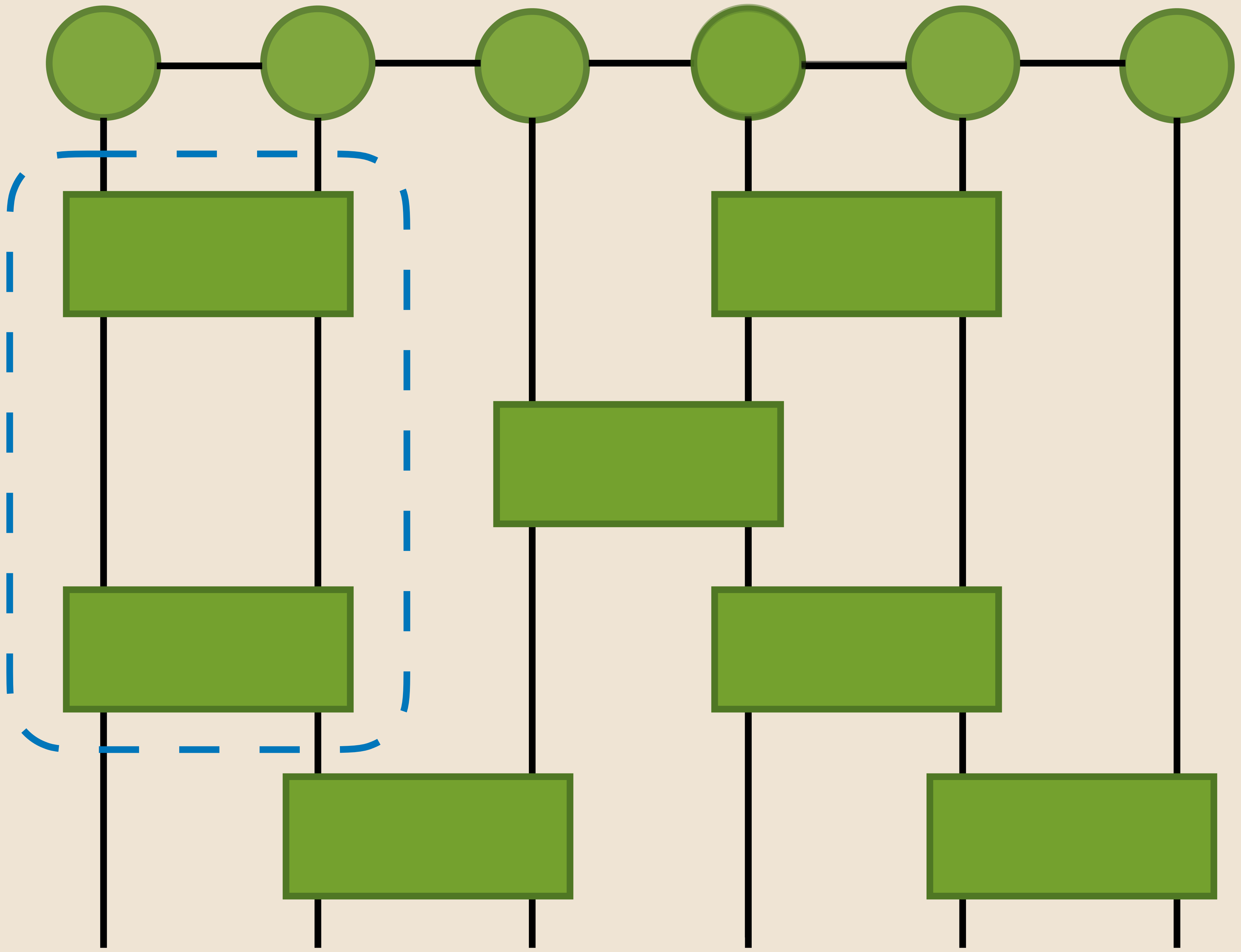
$$\mathcal{F}^{tot}(\chi) \geq \prod_i \mathcal{F}_i(\chi)$$

Optimisation & parallelism

Gates acts on the
same qubits:
we contract gates
together and only
after with state



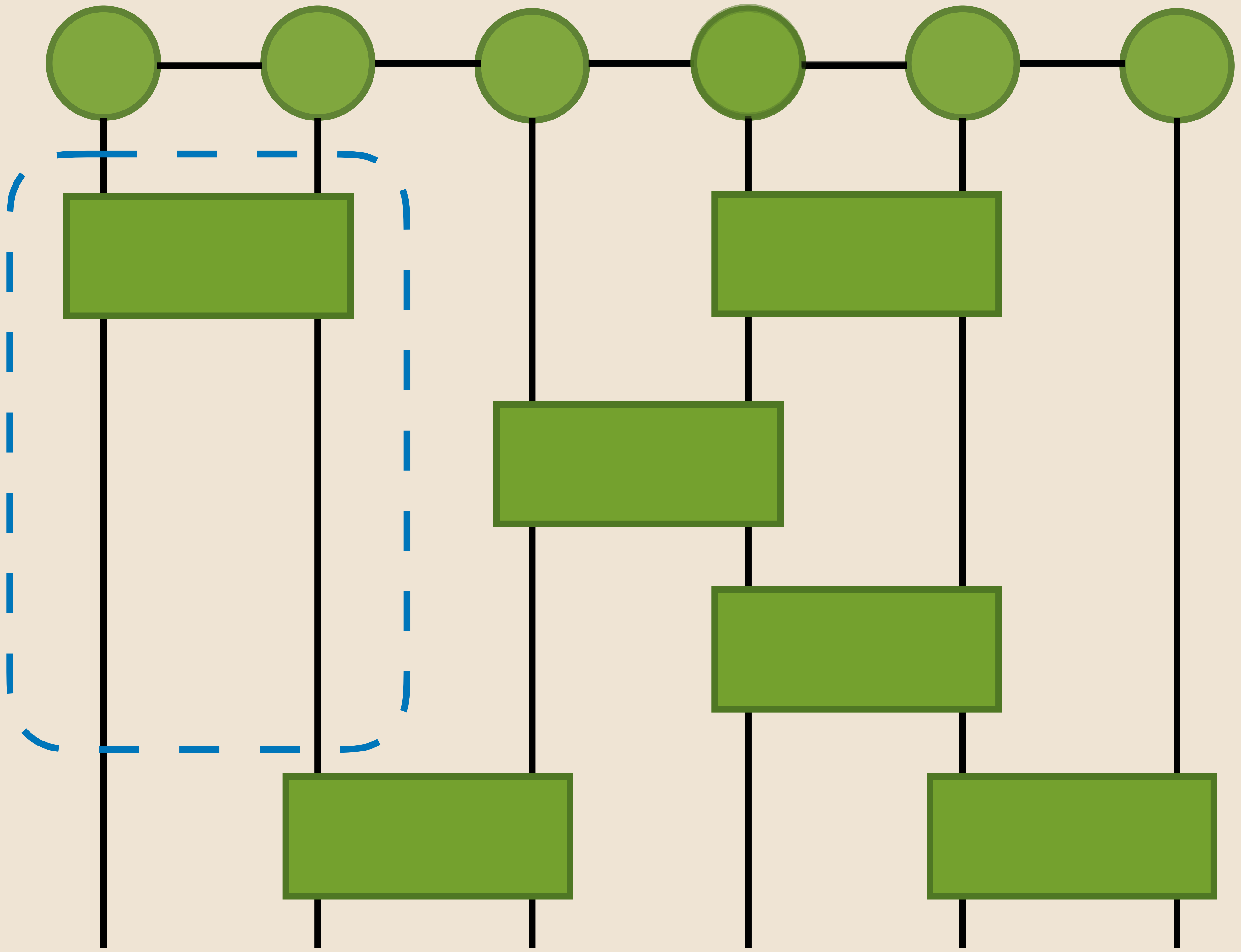
Optimisation & parallelism



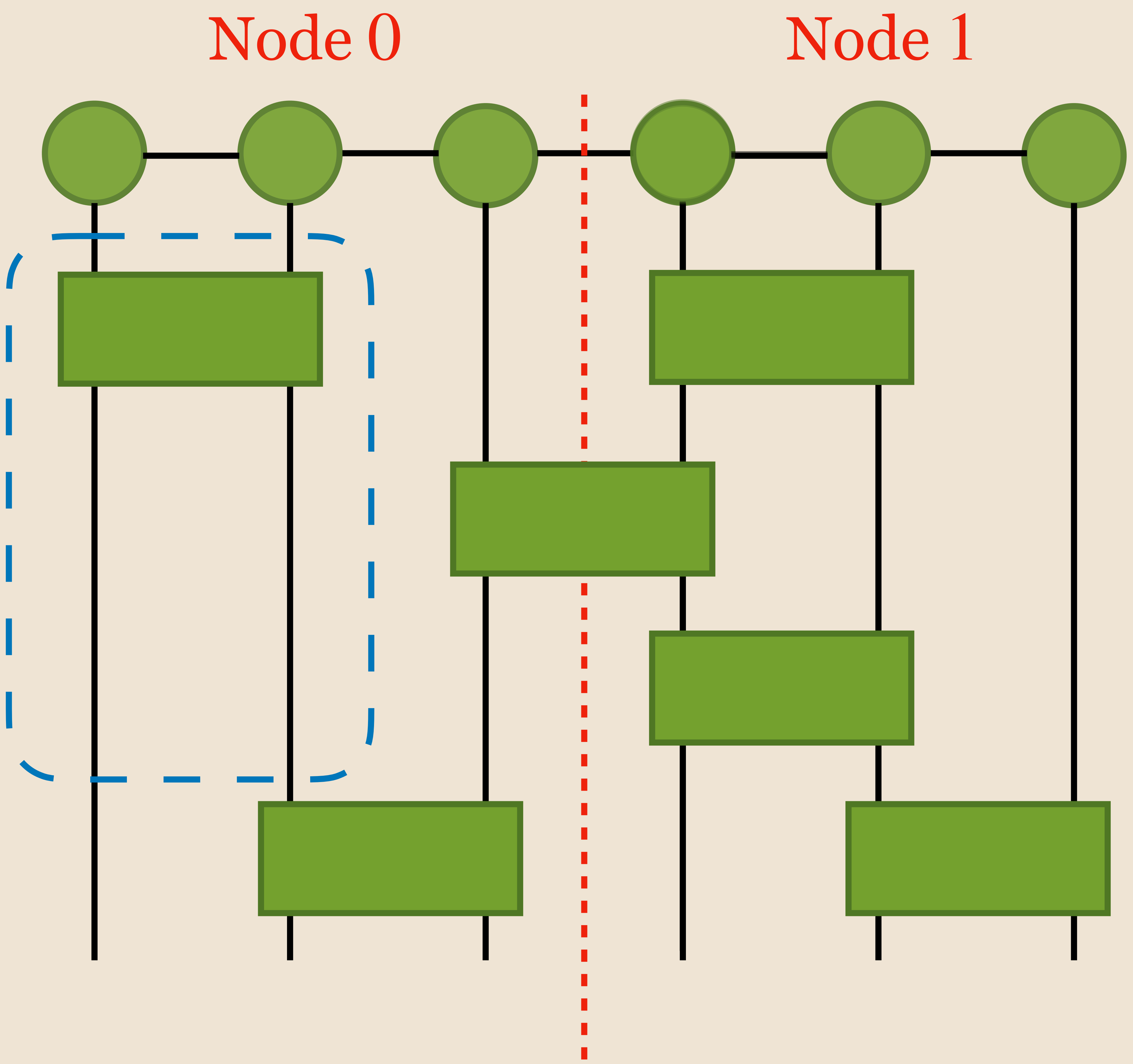
Gates acts on the same qubits: we contract gates together and only after with state

Optimisation & parallelism

Gates acts on the
same qubits:
we contract gates
together and only
after with state

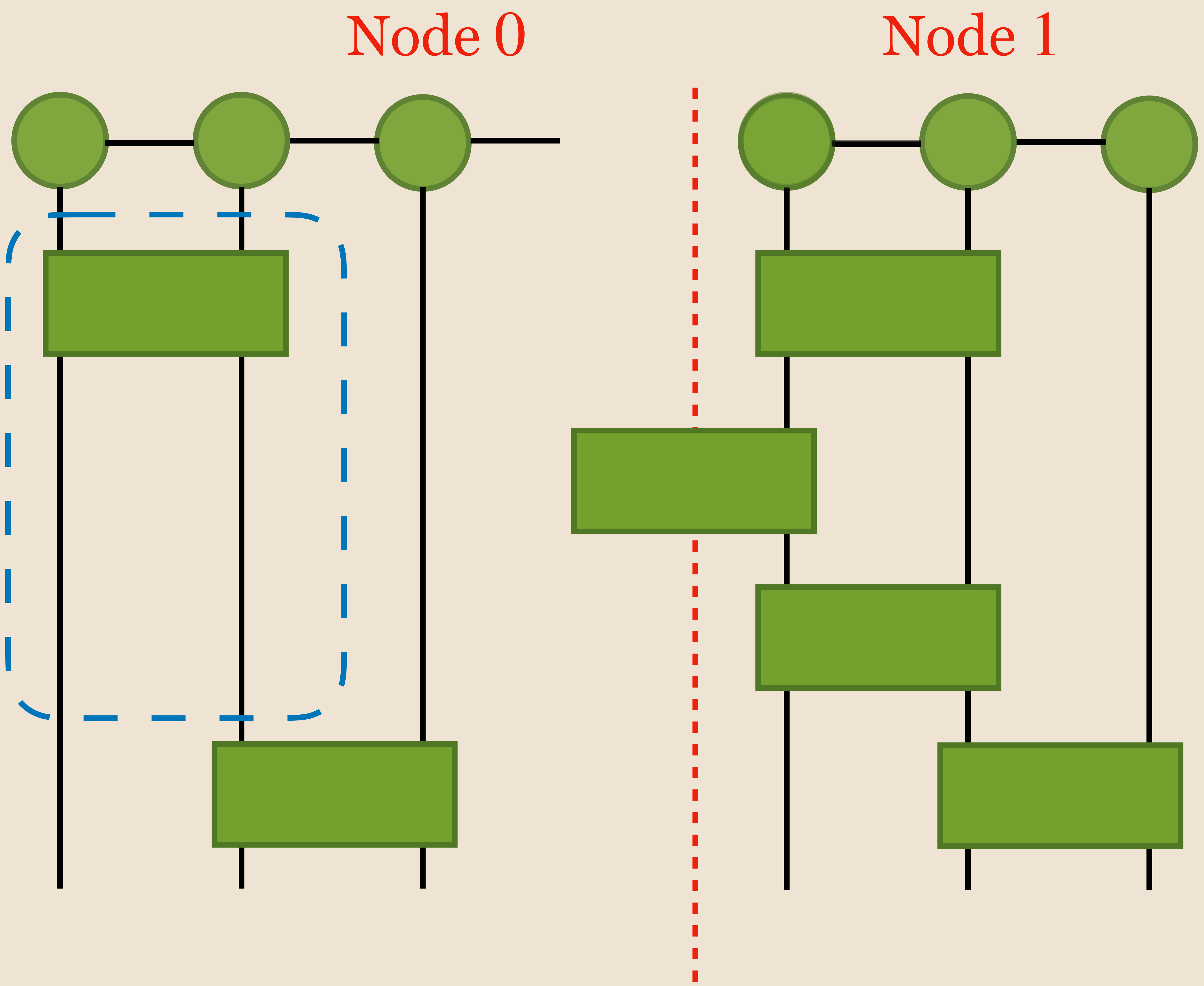


Optimisation & parallelism



Gates acts on the same qubits: we contract gates together and only after with state

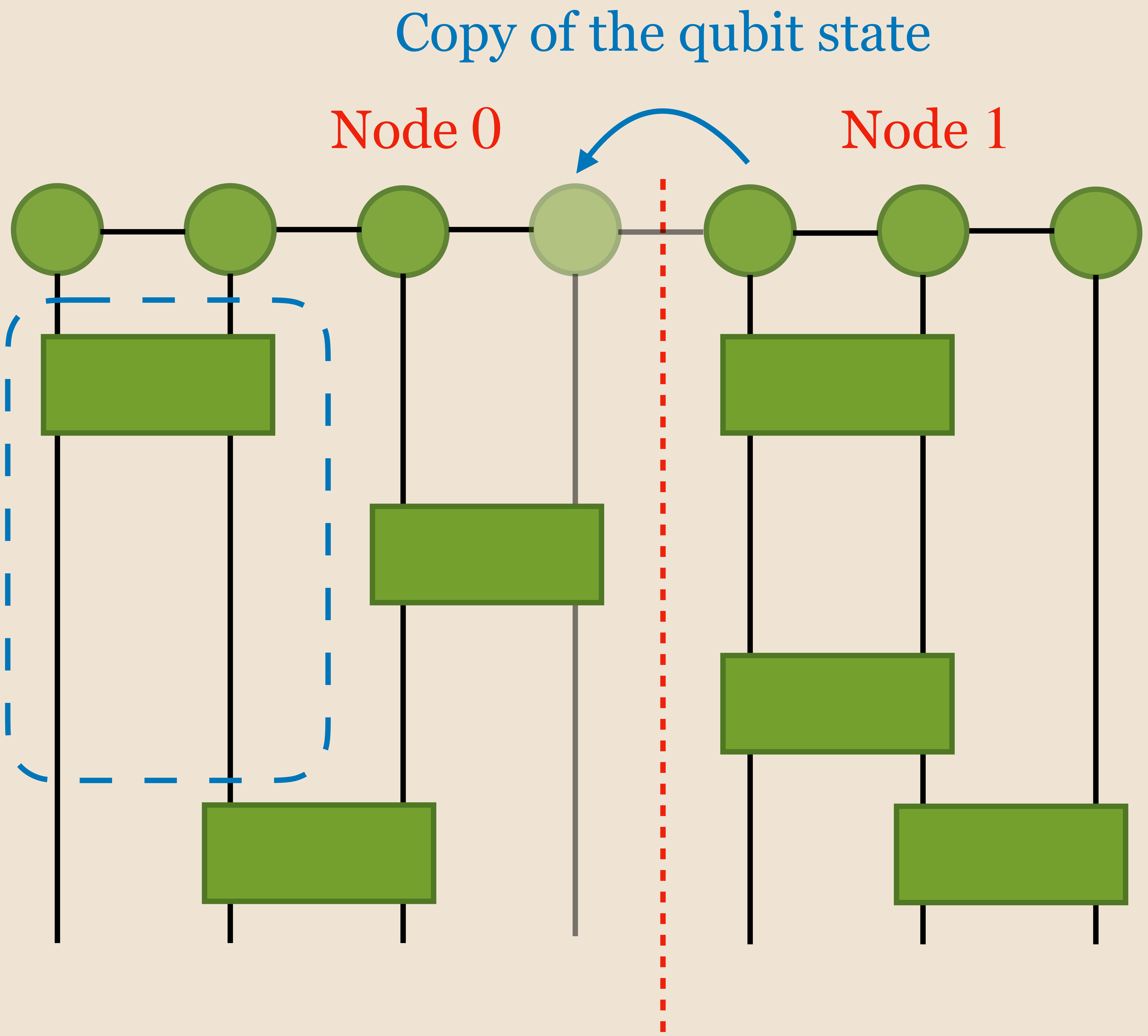
Optimisation & parallelism



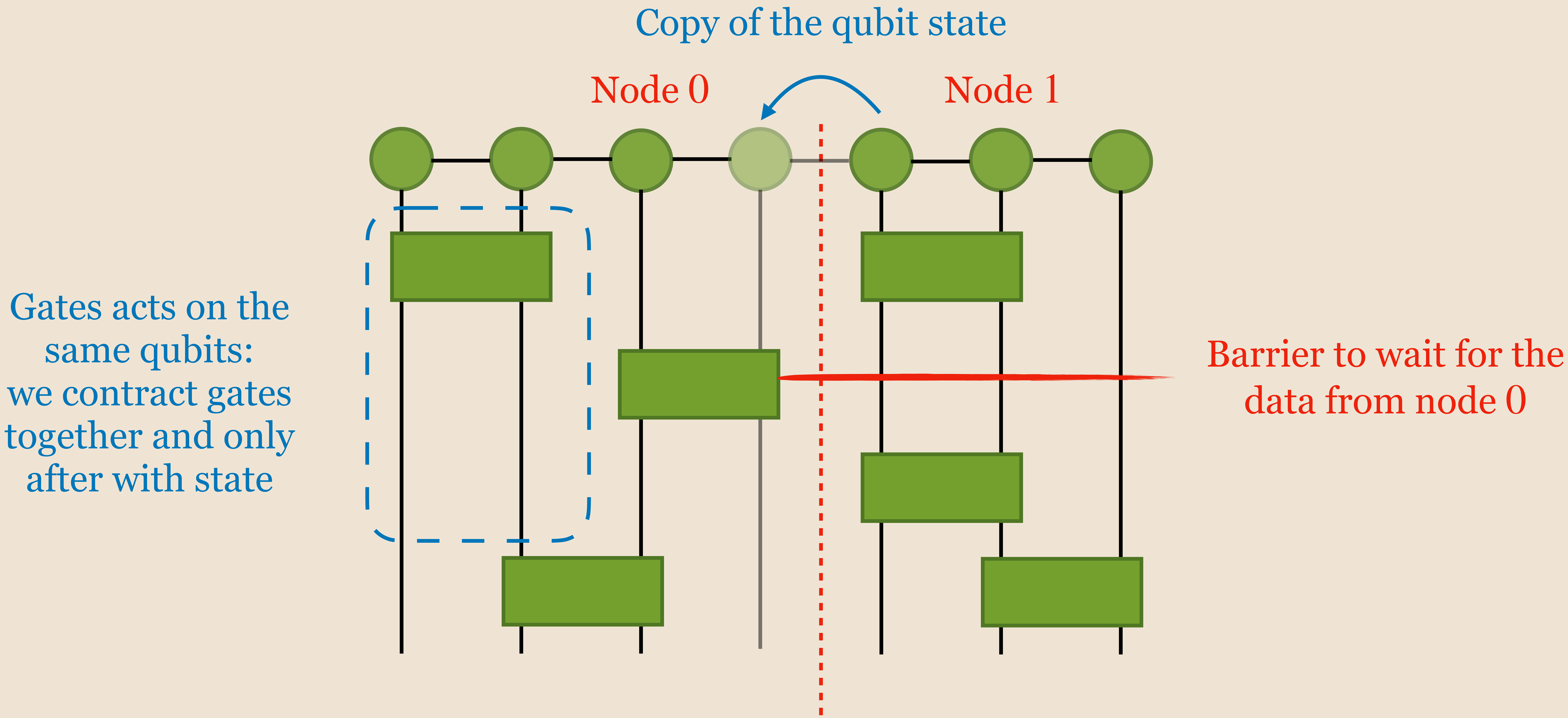
Gates acts on the same qubits: we contract gates together and only after with state

Optimisation & parallelism

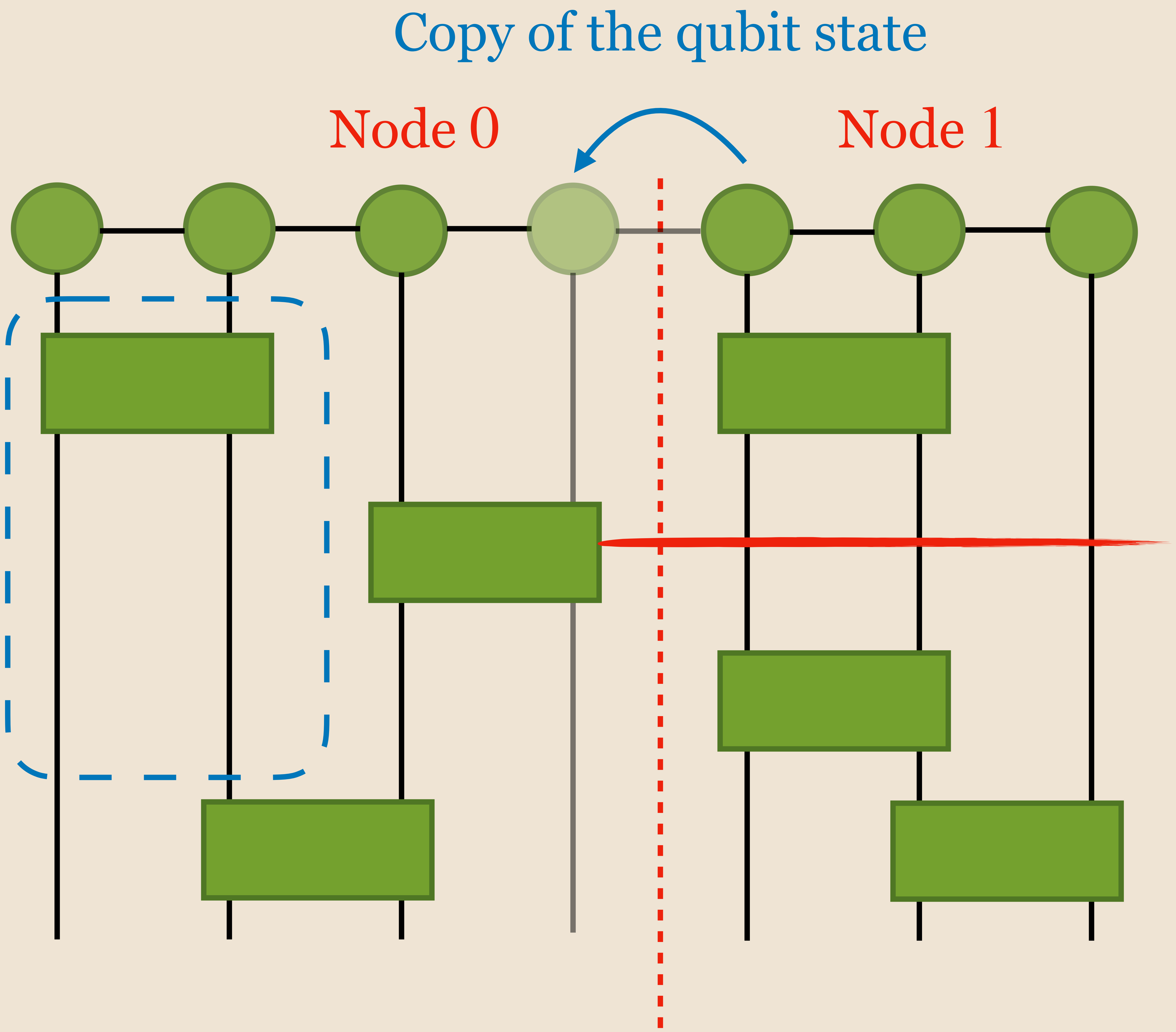
Gates acts on the same qubits: we contract gates together and only after with state



Optimisation & parallelism



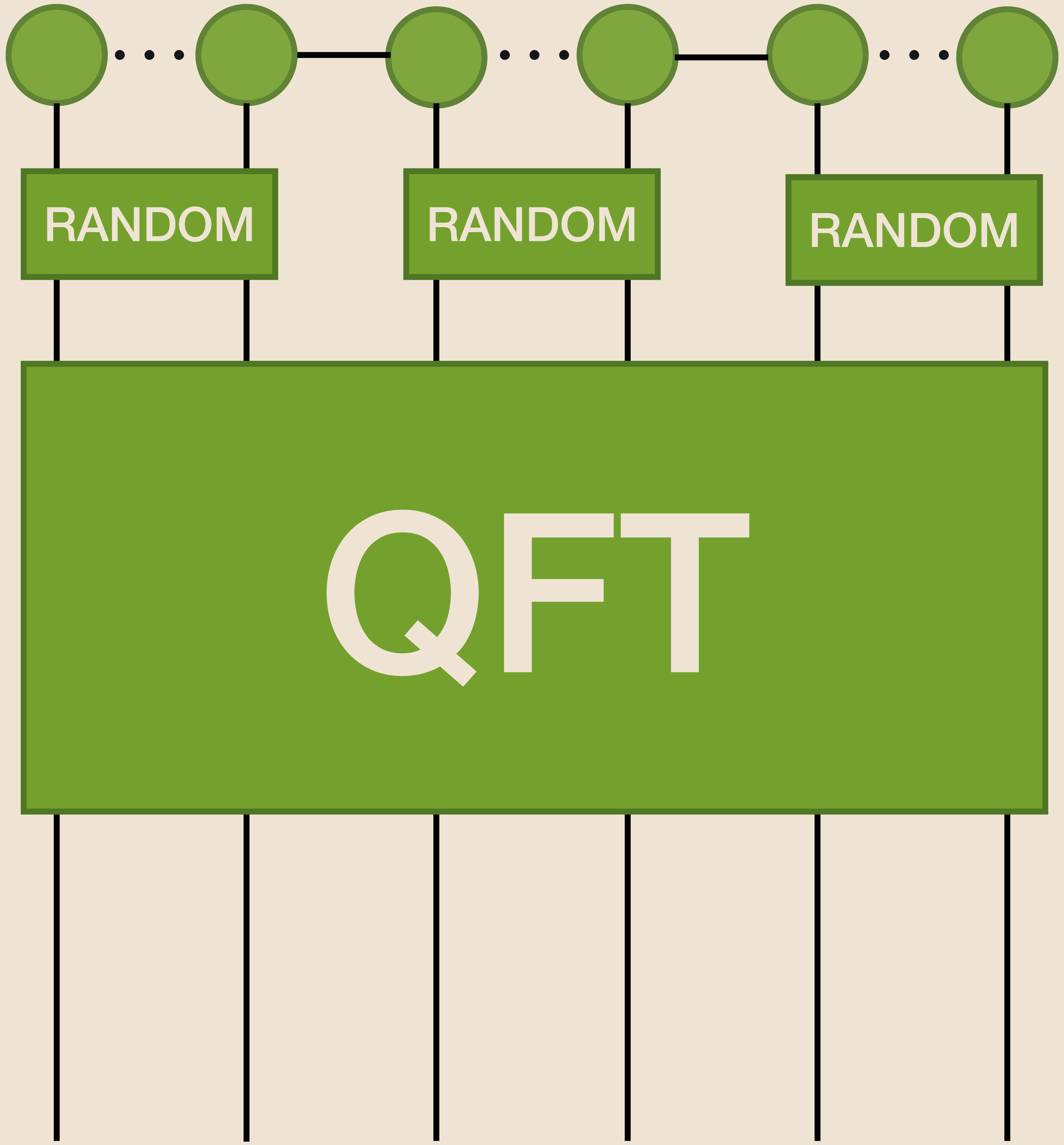
Optimisation & parallelism



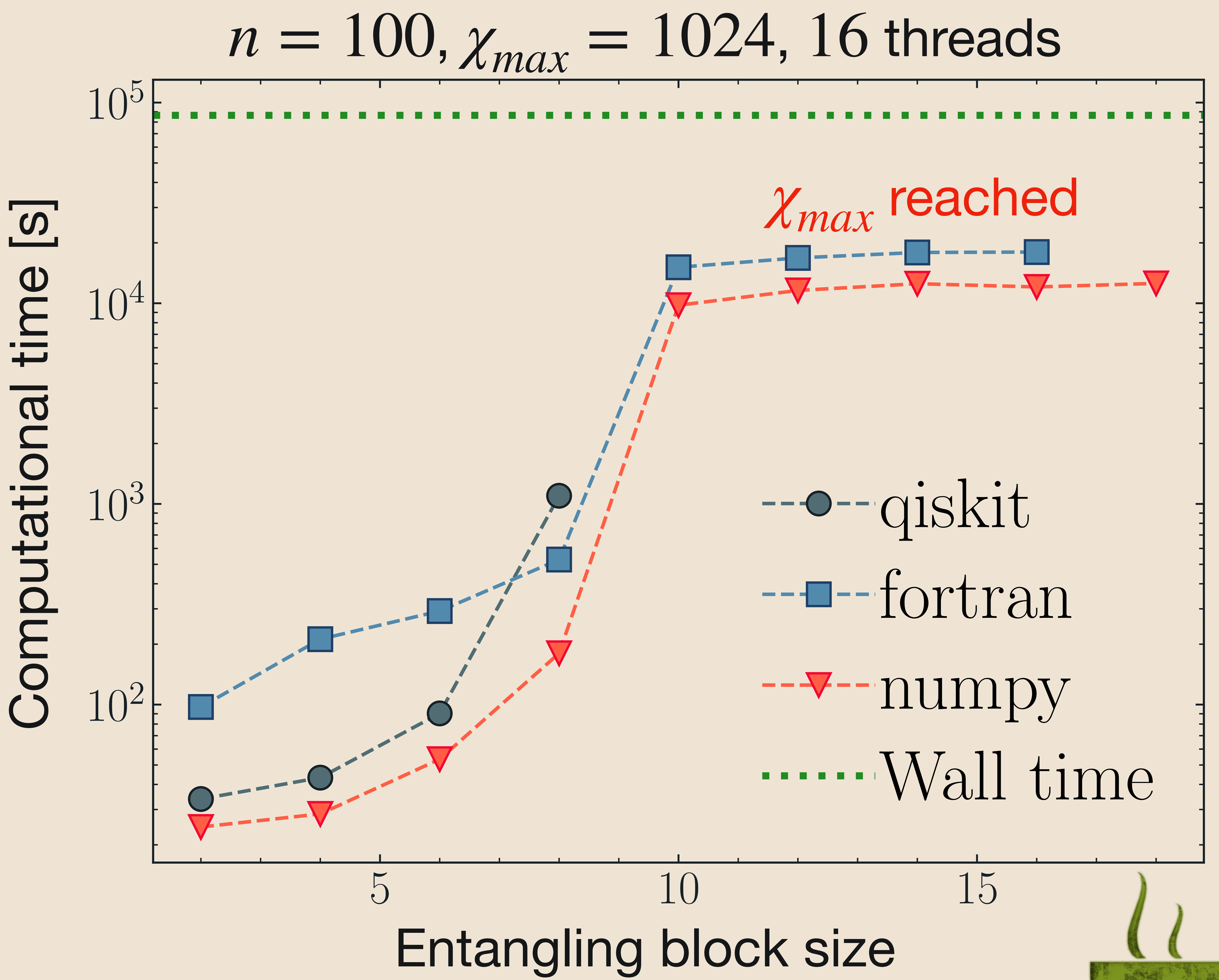
Gates acts on the same qubits: we contract gates together and only after with state

A GOOD PARALLEL SCALING INCREASES ERRORS DUE TO AN ALGORITHMIC SUBTLETY

Benchmarks

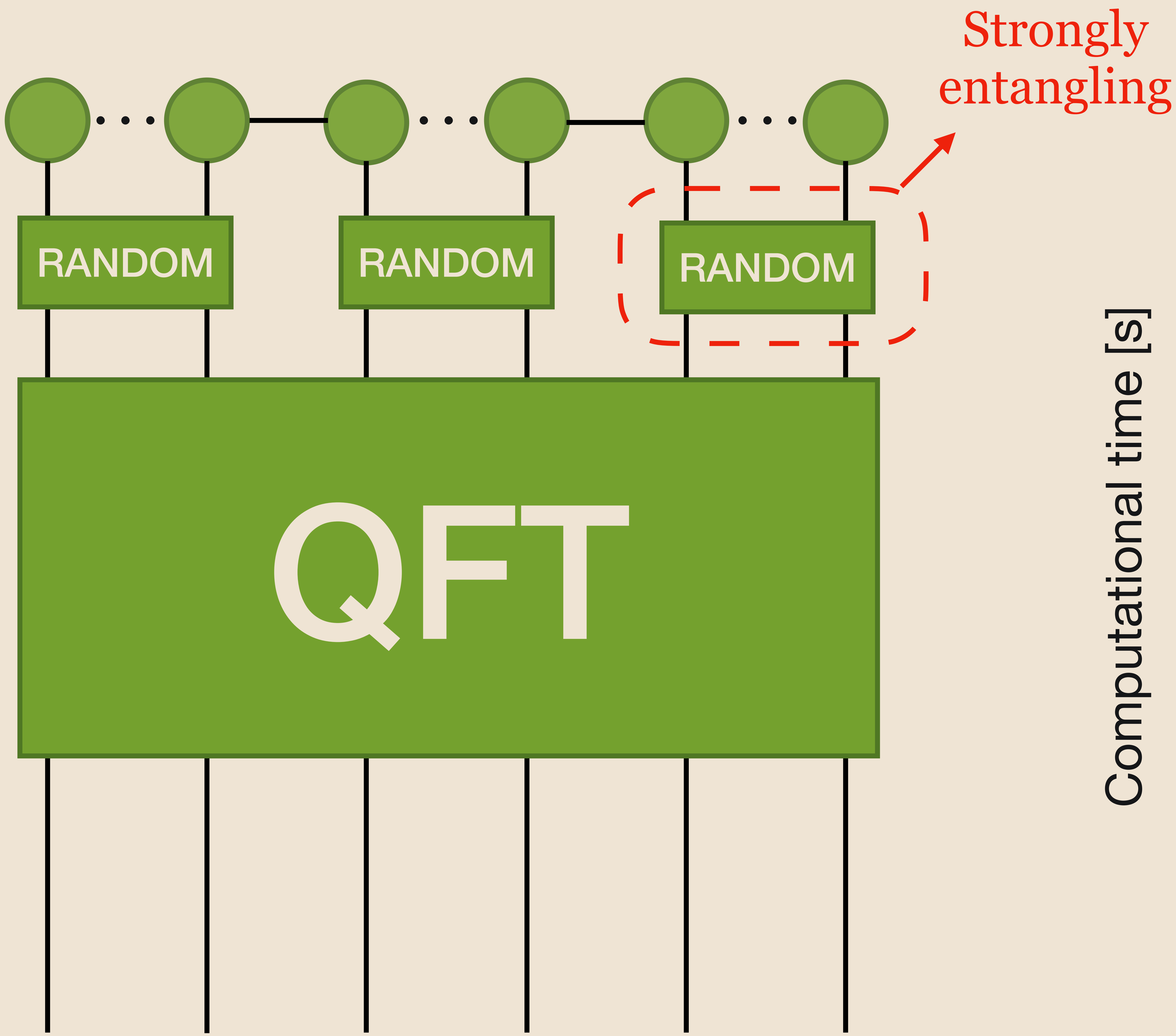


RANDOM
RANDOM
RANDOM
QFT

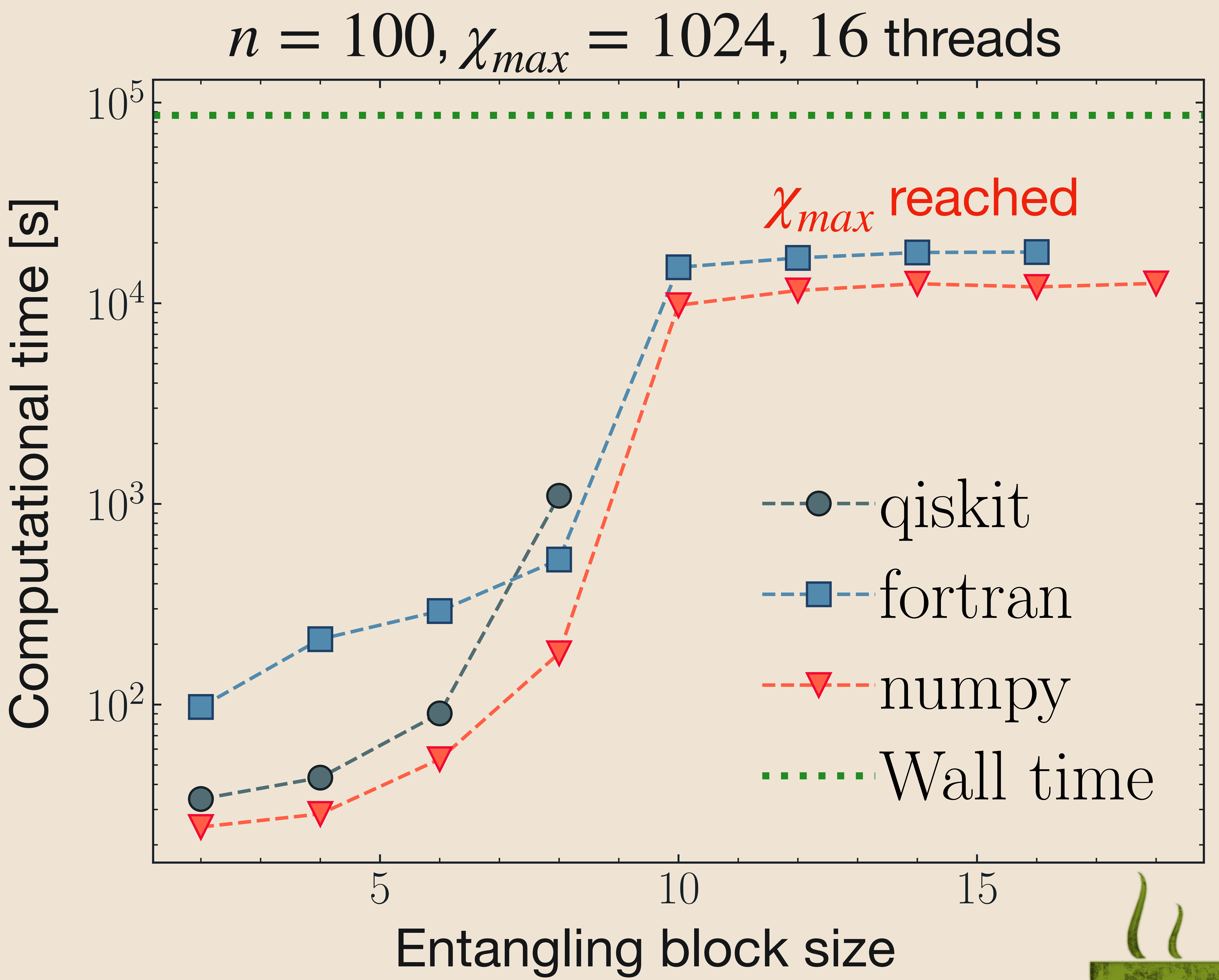


n = 100, χ_{max} = 1024, 16 threads
χ_{max} reached
Computational time [s]
Entangling block size
qiskit
fortran
numpy
Wall time

Benchmarks

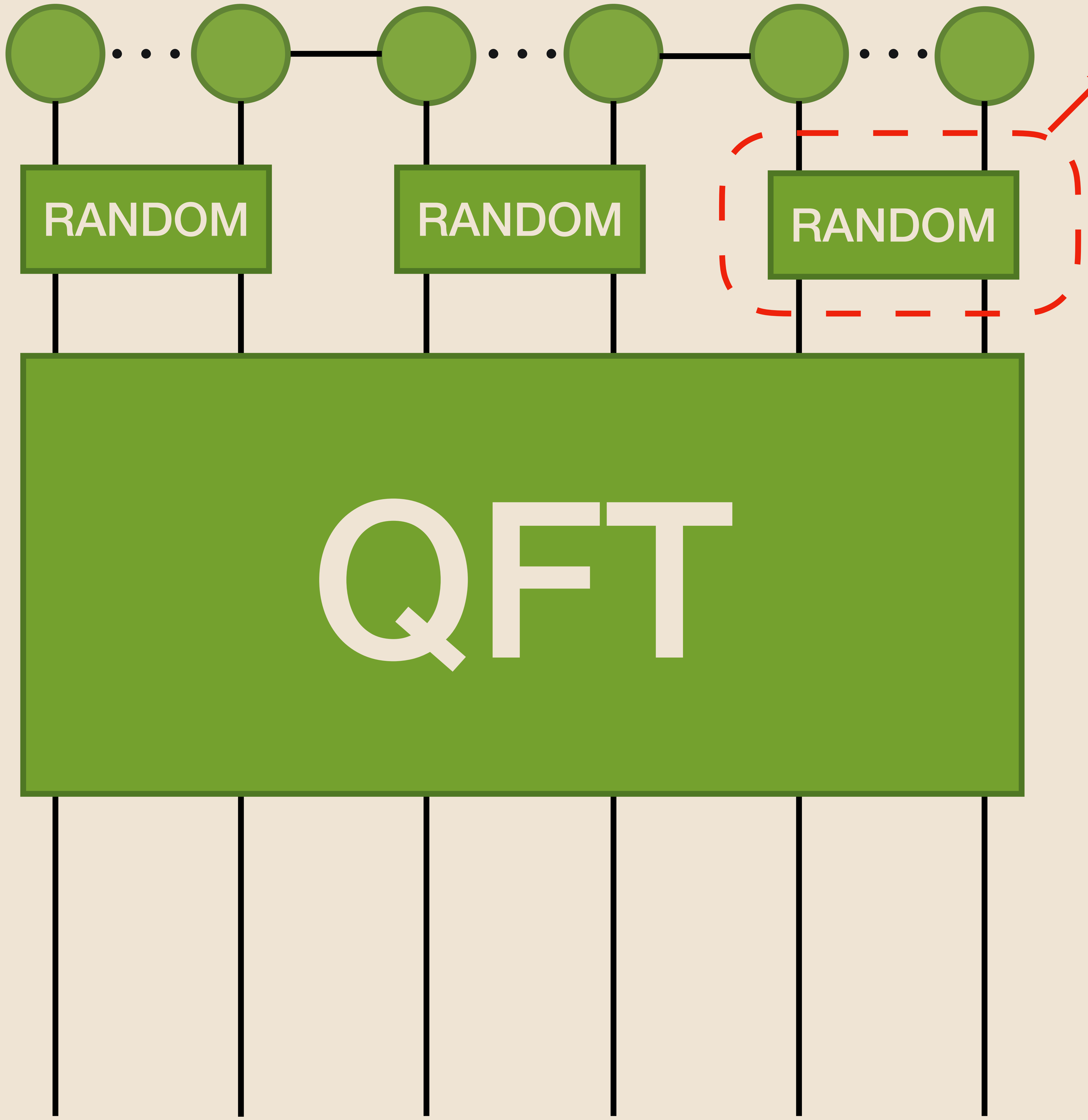


RANDOM
RANDOM
RANDOM
QFT
Strongly entangling

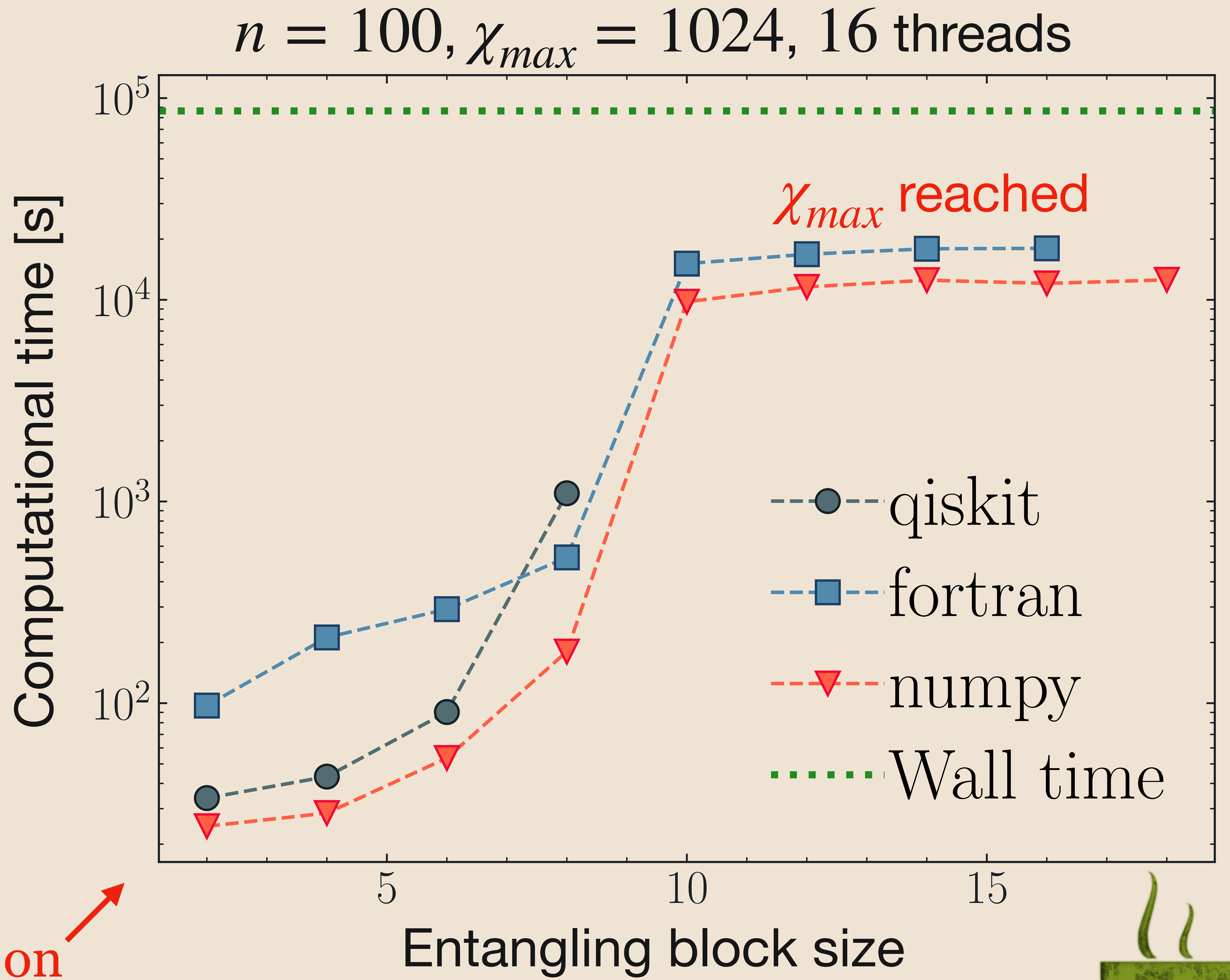


$n = 100, \chi_{max} = 1024$, 16 threads
Computational time [s]
Entangling block size
χ_{max} reached
qiskit
fortran
numpy
Wall time

Benchmarks



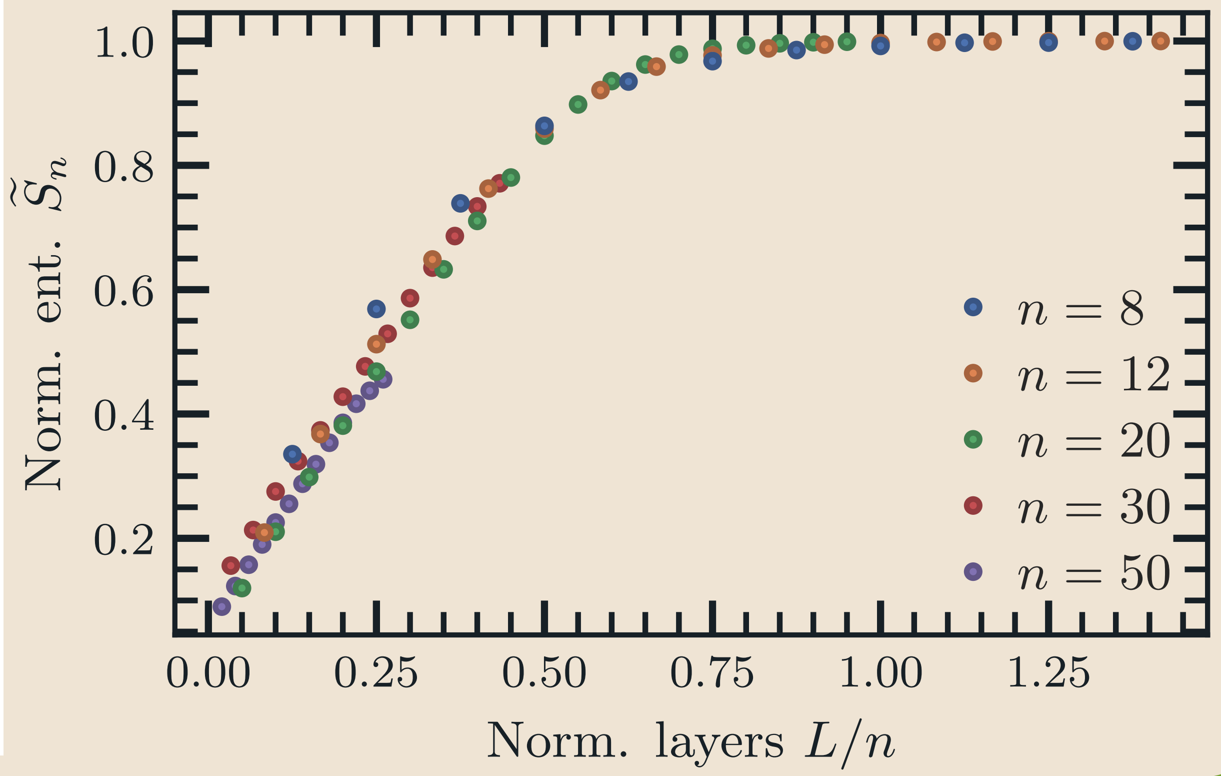
Strongly entangling



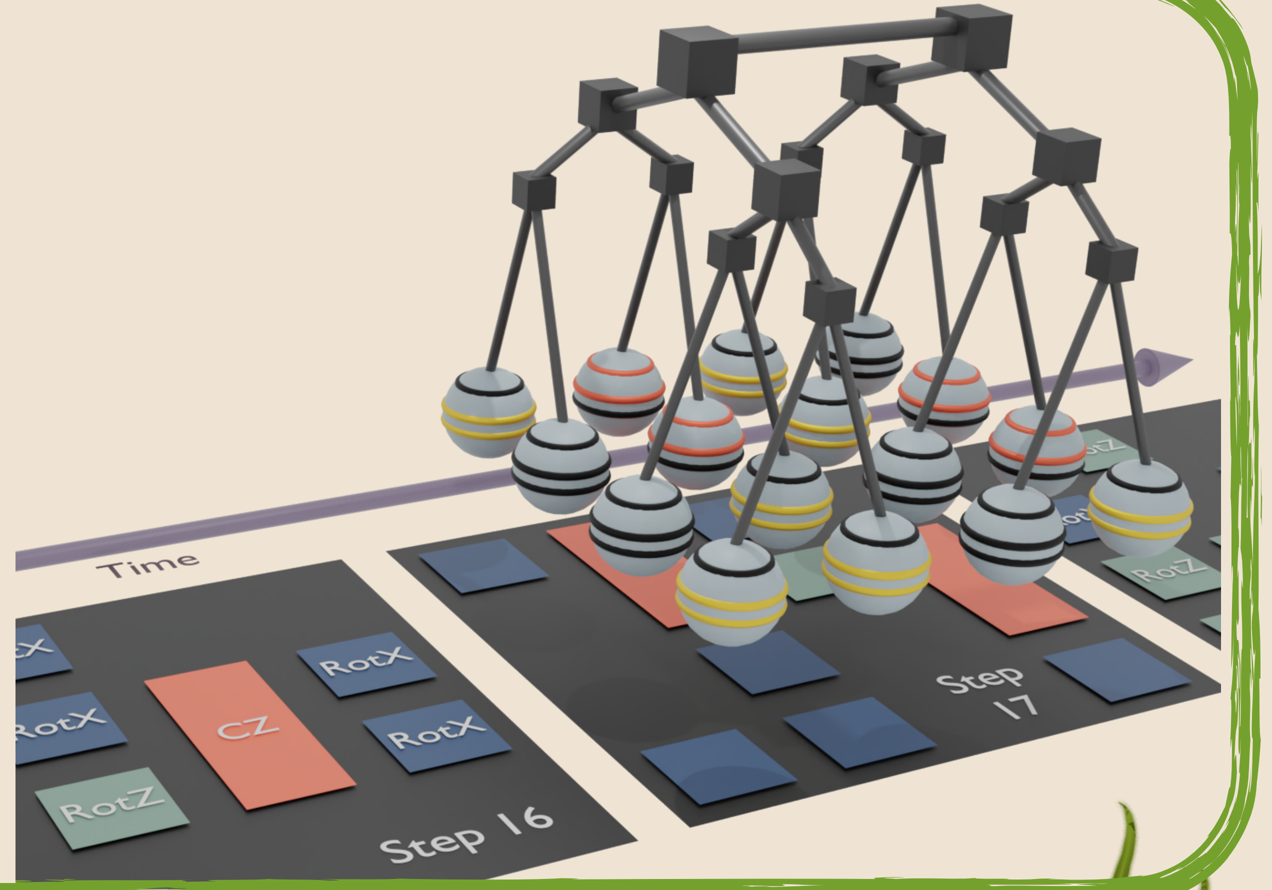
Run on Galileo100

Applications

Entanglement entropy production in QNN
Ballarin, Marco, et al. arXiv:2206.02474
- Simulations up to 50 qubits
- Bond dimension of 4096
- 11h of runtime on Galileo100

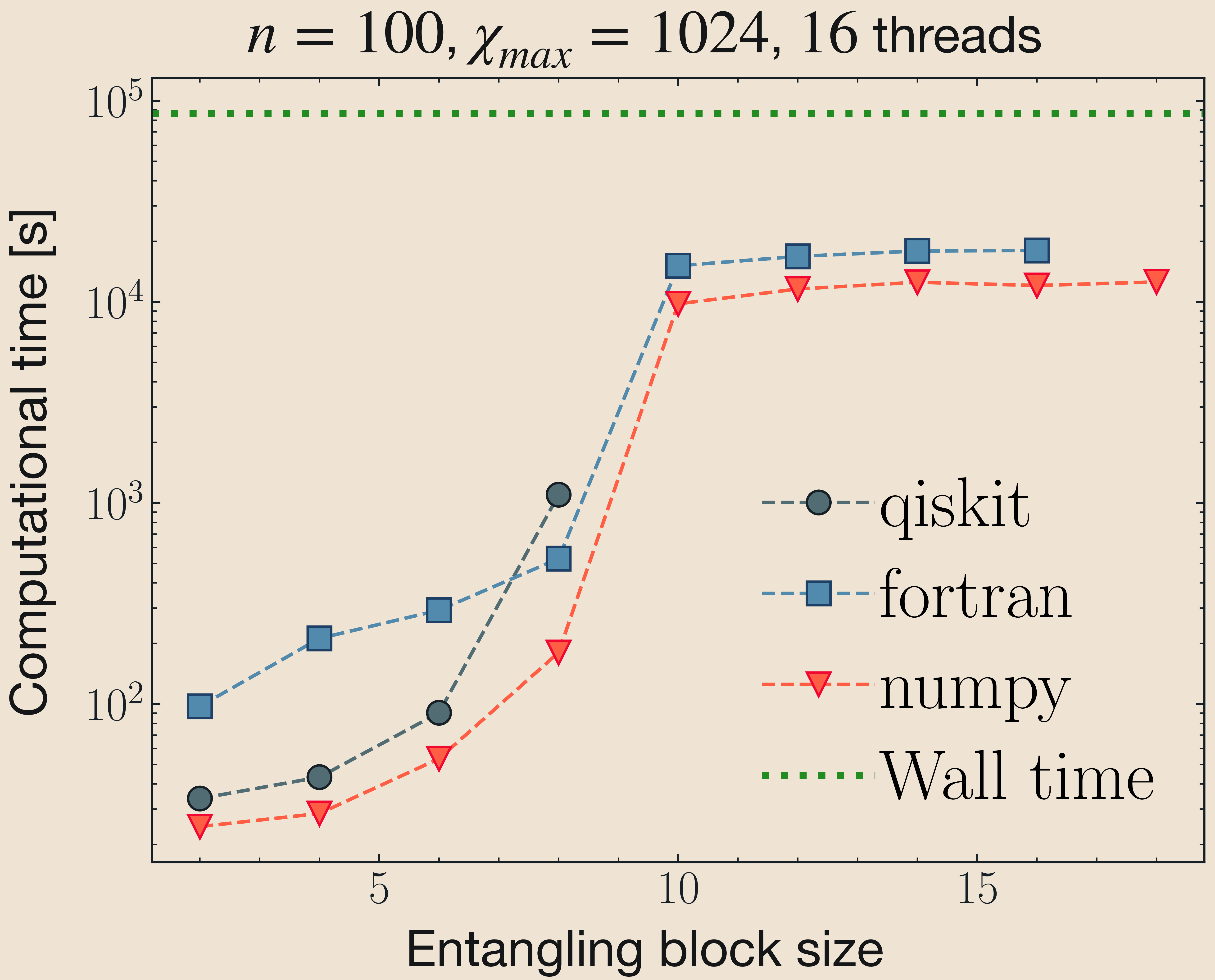


Ab initio two-dimensional digital twin for quantum computer
Jaschke, Daniel, et al. arXiv:2210.03763
- Use of the unbiased sampling
- Quantum matcha tea simulations used as target state to compute the fidelity of a simulation with crosstalk



Conclusions

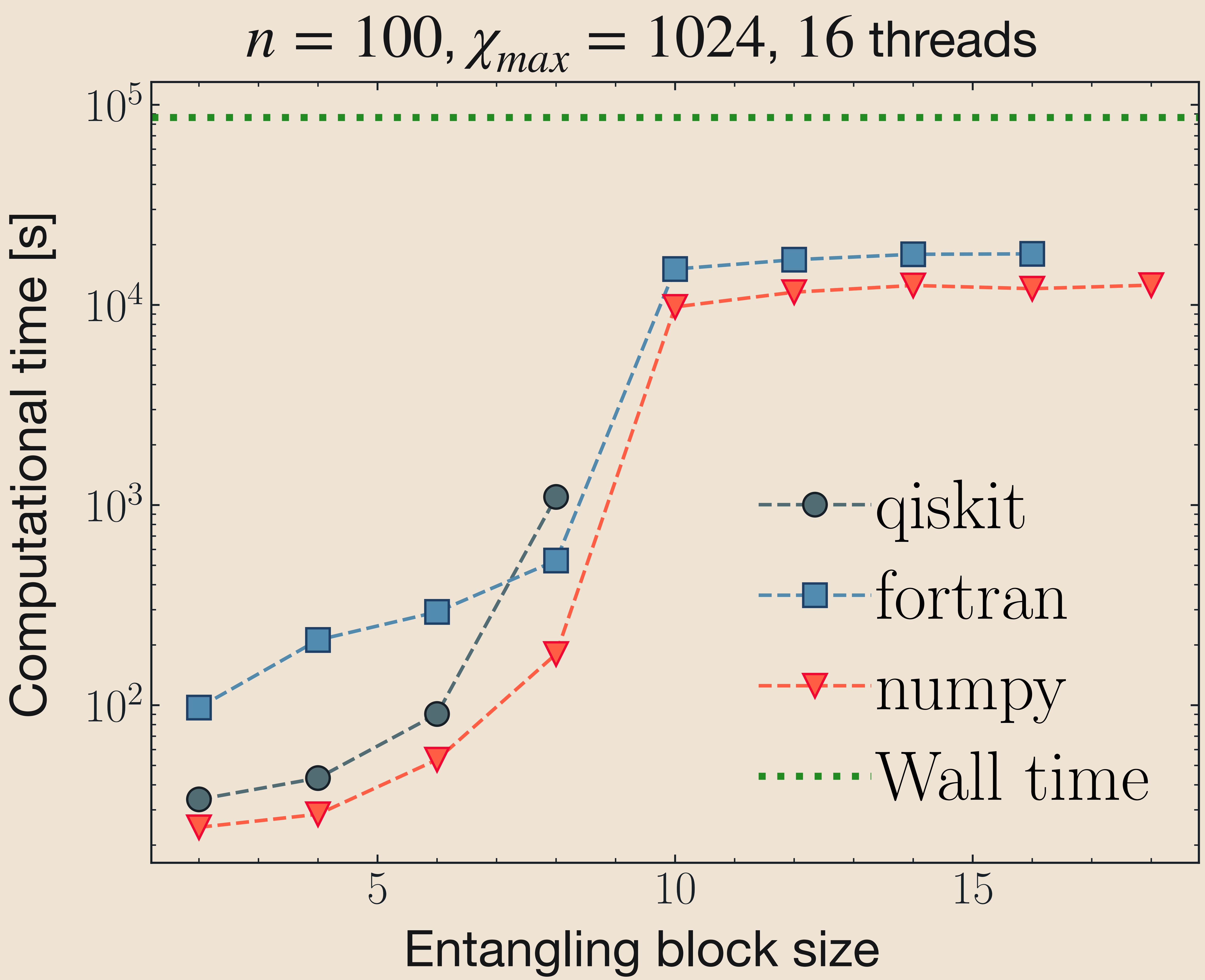
MPS simulations are not limited by the number of qubits but by the entanglement



Conclusions

MPS simulations are not limited by the number of qubits but by the entanglement

Easy-to-use python frontend and fast HPC-ready backend (Both GPU and CPU)

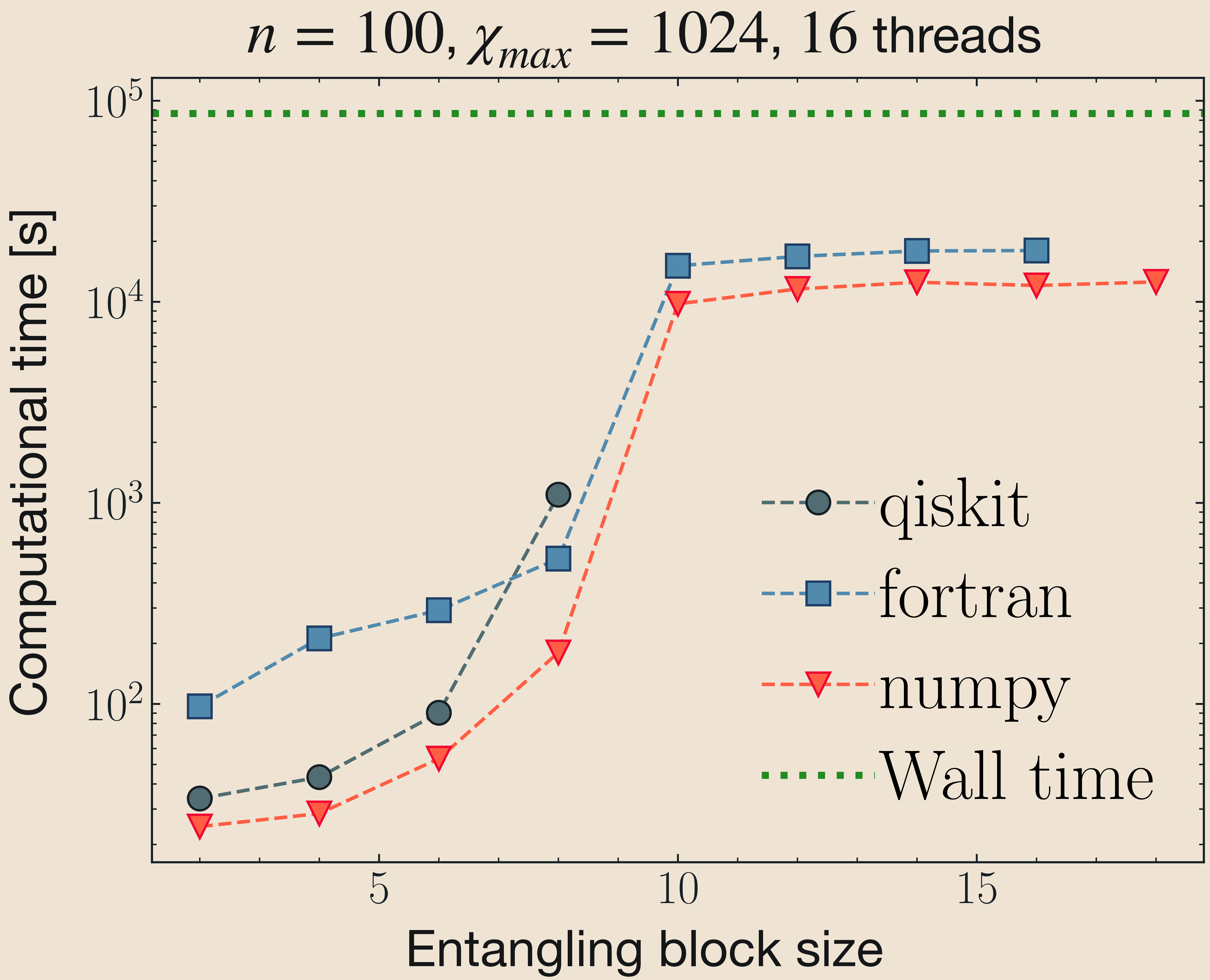


Conclusions

MPS simulations are not limited by the number of qubits but by the entanglement

Easy-to-use python frontend and fast HPC-ready backend (Both GPU and CPU)

Error analysis tools and efficient computations of observables optimised for the MPS representation



Thanks for your attention

Dipartimento
di Fisica
e Astronomia
Galileo Galilei

universität
uulm

CINECA

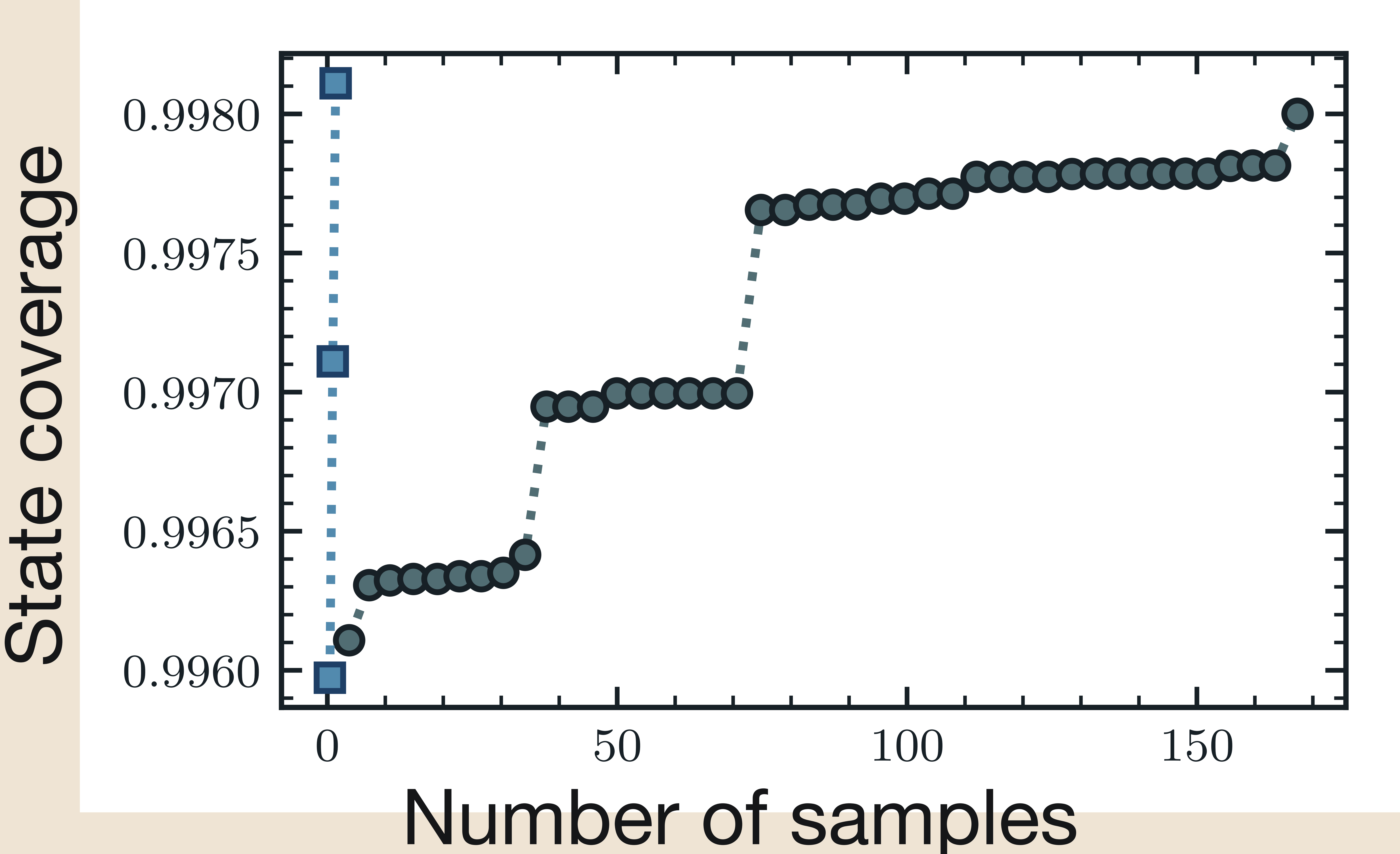

Simone Montangero


Daniel Jaschke


Riccardo Mengoni


Daniele Ottaviani

Efficient sampling of final state



Efficient sampling of final state